MEUHCUBE

API v1.2

# DESCRIPTION DU SYSTEME :

Le « *Meuhcube »* est conçu comme un carte d’extension pour l’Arduino Méga. Il est compatible avec les autres cartes Arduino partageant le même pinout notamment l’Arduino Due (NDLA : à vérifier).

Il comporte 343 LEDS assemblées en un cube de 7 LEDS d’arrête. Toutes les LEDS d’une même colonne partagent la même anode et toutes les LEDS d’un même étage partagent la même cathode. Les étages du cube sont allumés en séquence de manière à provoquer l’illusion d’une image continue.

L’intervalle entre l’allumage de chaque couche est défini par le paramètre « interlayer », il est défini par défaut à 1428 µs ce qui permet un rafraichissement général de 60 Hz.

L’état des LEDS est rafraichit en temps mais il est également possible d’utiliser le paramètre « interframe » (NDLA : pas encore implémenté) pour qu’il soit synchronisé sur une horloge interne. Il est également possible de le piloter via une horloge externe grâce au paramètre « extclock ».

# BROCHES d’entrée/sortie :

Schéma du pinout

# LE SYSTEME DE COORDONNÉES :

# IDE DISPONIBLE :

Arduino, Processing, Max/MSP (javascript), Python

Autre environnements uilisables : Labview, Puredata, NodeJS, le Lab View de Luc, Open FrameWorks

# Laison série :

Le liaison série est étable via le port virtuel USB de l’Arduino Méga, la communication est établie à 115200 bauds, 8bits de données, 1 bit de stop, pas de parité.

Dans les prochaines versions, il sera également possible de communiquer via les ports séries auxiliaires de l’Arduino Méga. Cela donne la possibilité d’ajouter un second microcontrôleur chargé de préparer les animations ou des liaisons sans fil grâce à des modules xBee, Bluetooth ou Wifi.

# Structure des paquets :

Les paquets sont composés de trois octets d’entête, des données (DATA) et d’un octet de fin de paquet (EOP). L’entête se compose, quant à elle, d’un octet de début de paquet (SOP), d’un octet de description de la commande (CMD) et d’un octet spécifiant la taille des données envoyées (LEN).

SOP CMD LEN (DATA) EOP

SOP valeur constante définie à 0xE7

CMD la commande à exécuter

LEN la longueur des données transmises

DATA les données transmises (optionnel)

EOP valeur constante définie à 0x7E

Ex : 0xE7 0x00 0x01 0x00 0x7E

# Structure des COMMANDES :

Chaque commande est identifiée par un numéro de commande sur un octet. La longueur des données associées aux commande définissent la fonction qui sera appliquée.

## ALLUMER/ETEINDRE DES LEDS

### 0x00 - SET\_ALL

|  |  |  |  |
| --- | --- | --- | --- |
| Commande: | CMD | LEN | Registre d’état |
| 0x00 | 0x01 | [ 0x00 -> 0x7F] |

Explication Data:

Il faut penser en binaire avec 1 bit égal une couche horizontal (registre d’état), donc :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| BIT7 | BIT6 | BIT5 | BIT4 | BIT3 | BIT2 | BIT1 | BIT0 |
| Non utilisé | Layer-7-Enable | Layer-6-Enable | Layer-5-Enable | Layer-4-Enable | Layer-3-Enable | Layer-2-Enable | Layer-1-Enable |

0x00= 00000000 -> 7 couches éteintes,

0x20= 00100000 -> couche 6 allumée,

0x24= 00100100 -> couche 6 et 3 allumées,

0x80= 10000000 -> 7 couches éteintes,

### 0x01 – SET\_PIXEL

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | | | |
| CMD | LEN | X | Y | Z | ETAT |
| 0x01 | 0x04 | [ 0x00 -> 0x06] | [ 0x00 -> 0x06] | [ 0x00 -> 0x06] | [ 0x00 -> 0x01] |

On donne la valeur des coordonnées cartésiennes de la LED à modifier, et son état.

### 0x02 – SET\_LINE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | | | |
| CMD | LEN | Axe de référence | Coordonnée 1 | Coordonnée 2 | Registre d’état |
| 0x02 | 0x04 | [ 0x00 -> 0x02] | [ 0x00 -> 0x06] | [ 0x00 -> 0x06] | [ 0x00 -> 0x7F] |

L’axe de référence permet de choisir l’axe perpendiculaire au plan dans lequel Coordonnée 1 et Cordonnée 2 sont définies

Le registre d’état permet de définir l’état des 7 LED de la ligne sélectionnée.

Axe de référence :

0x00= Plan de gauche de Meuh cube

(0;0)

0x01= Plan de Meuh cube

(0;0)

0x02 = Plan de dessus du cube

(0;0)

### 0x03 – SET\_PLAN

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | | Données | | |
| CMD | LEN | Plan de référence | | Coordonnée | Registre d’état |
| 0x03 | 0x03 | [ 0x00 -> 0x02] | | [ 0x00 -> 0x06] | [ 0x00 -> 0x7F] |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | | | |
| CMD | LEN | Plan de référence | Coordonnée | Registre d’état | Registre d’état |
| 0x03 | 0x09 | [ 0x00 -> 0x02] | [ 0x00 -> 0x06] | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Registre d’état | Registre d’état | Registre d’état | Registre d’état | Registre d’état |
| [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] |

Plan de référence :

0x00 : Plan de Meuh cube

0x01 : Plan de dessous du cube

0x02 : Plan de gauche de Meuh cube

Coordonnée = éloignement de façon parallèle au plan de référence

Le registre d’état permet de définir l’état des 49 LED en définissant une ligne et en la propageant sur le ligne sélectionnée selon l’axe complémentaire.

Les registres d’état permettent de définir l’état des 49 LED en définissant ligne à ligne le plan sélectionnée.

### 0x04 – SET\_CUBE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | | Données | | |
| CMD | LEN | Registre d’état | | Registre d’état | Registre d’état |
| 0x04 | 0x31 | [ 0x00 -> 0x7F] | | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] |

49 Registres d’état qui gère l’affichage des 49 colonnes du cube à la verticale.

## INVERSER L’ETAT DES LEDS

### 0x10 - REV\_ALL

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 010 | 0x00 |

### 0x11 – REV\_PIXEL

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | | Données | | |
| CMD | LEN | X | | Y | Z |
| 011 | 0x03 | [ 0x00 -> 0x7F] | | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] |

### 0x12 – REV\_LINE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | | | |
| CMD | LEN | Axe de référence | Coordonnée 1 | Coordonnée 2 | Registre d’état |
| 0x12 | 0x04 | [ 0x00 -> 0x02] | [ 0x00 -> 0x06] | [ 0x00 -> 0x06] | [ 0x00 -> 0x7F] |

### 0x13 – REV\_PLAN

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | | Données | | |
| CMD | LEN | Plan de référence | | Coordonnée | Registre d’état |
| 0x13 | 0x03 | [ 0x00 -> 0x02] | | [ 0x00 -> 0x06] | [ 0x00 -> 0x7F] |

### 0x14 – REV\_CUBE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Commande: | Description commande | | | Données | | |
| CMD | LEN | Registre d’état | | Registre d’état | Registre d’état |
| 0x14 | 0x31 | [ 0x00 -> 0x7F] | | [ 0x00 -> 0x7F] | [ 0x00 -> 0x7F] |

## ANIMER LE CUBE

### 0x21 – MOVE\_XYZ

bouger l'ensemble des pixels dans l'axe choisi (trouver un moyen mémotechnique pour que ce soit facile à retenir, comme pour les lignes

|  |  |  |  |
| --- | --- | --- | --- |
| Commande: | Description commande | | Données |
| CMD | LEN | Axe du mouvement |
| 0x21 | 0x01 | [ 0x00 -> 0x05] |

Mouvement de 1 cran

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | |
| CMD | LEN | Axe du mouvement | Nombre de pas |
| 0x21 | 0x02 | [ 0x00 -> 0x05] | [ 0x00 -> 0x07] |

Mouvement de x crans

### 0x22 – MODU\_XYZ

mouvement torique (bouclage)

|  |  |  |  |
| --- | --- | --- | --- |
| Commande: | Description commande | | Données |
| CMD | LEN | Axe du mouvement |
| 0x22 | 0x01 | [ 0x00 -> 0x05] |

Mouvement de 1 cran

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | |
| CMD | LEN | Axe du mouvement | Nombre de pas |
| 0x22 | 0x02 | [ 0x00 -> 0x05] | [ 0x00 -> 0x07] |

Mouvement de x crans

### 0x23 – ROTA\_XYZ

rotation du plan

### 0x24 – MIRO\_XYZ

mirroir sur un axe

## COMMANDES SYSTEMES

### 0xE0 – VERSION

get version number

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 0xE0 | 0x00 |

### 0xE1 – DEFAULT

load default parameters

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 0xE1 | 0x00 |

### 0xE2 – LOAD

load last saved parameters

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 0xE2 | 0x00 |

### 0xE3 – SAVE

save current parameters

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 0xE3 | 0x00 |

## PARAMETRES SYSTEMES

### 0xF0 – EXTCLOCK

set external clock

|  |  |  |  |
| --- | --- | --- | --- |
| Commande: | Description commande | | Données |
| CMD | LEN | Booleen qui autorise l’utilisation d’une horloge externe |
| 0xF0 | 0x01 | [ 0x00 -> 0x01] |

### 0xF1 – FRAMERATE

define interFrame value

|  |  |  |  |
| --- | --- | --- | --- |
| Commande: | Description commande | | Données |
| CMD | LEN | Interval ms entre deux rafraichissement du cube |
| 0xF1 | 0x01 | [ 0x00 -> 0xFF] |

### 0xF2 – LAYERRATE

define interLayer value

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Commande: | Description commande | | Données | |
| CMD | LEN | MSB – bit 15 à 8 | LSB – Bit 7 à 0 |
| 0xF2 | 0x02 | [ 0x00 -> 0xFF] | [ 0x00 -> 0xFF] |

### 0xFF – REFRESH

refresh frame

|  |  |  |
| --- | --- | --- |
| Commande: | Description commande | |
| CMD | LEN |
| 0xFF | 0x00 |