CH-1092 Belmont, Switzerland
Tel +41 21 728-6156, Fax 728-6157
Email info@didel.com, www.didel.com

Edu-C Graphique avec le Oled I2C SSD1306

- version provisoire -

1 Introduction

Vous avez un Edu-C et vous savez ce que l'on peut faire de simple avec la librairie Oled.h.

L'affichage Oled SSD1306 est très utile pour visualiser des signaux et remplacer le terminal Arduino, avec l'avantage de faire partie de l'application contrairement au terminal série.

2 Ordres vus dans Edu-C

2.1 Afficher des nombres

Les fonctions à disposition sont

Licol(li,co);	. Place le pointeur ligne li (0 à 7) et en colonne co (0 à 127, mais il faut la place pour finir)
Bin8(v8);	. Affiche la variable 8 bits (byte, char, int8_t, uint8_t) en binaire
Hex8(v8);	. Affiche la variable 8 bits (byte, char, int8_t, uint8_t) en hexadécimal
Hex16(v16);	. Affiche la variable 16 bits (int, int16_t uint6_t) en hexadécimal
Dec8(v8);	. Affiche la variable 8 bits (byte, char, int8_t, uint8_t) en décimal
Dec16(v16);	. Affiche la variable 16 bits (int, int16_t uint6_t) en décimal, limite 0xFFFF=65535
Dec9999(v12);	. Affiche 4 digits. C'est rare que l'on doive afficher des nombres plus grands que 9999 = 16'270F en interagissant avec des capteurs. L'écran est petit, un chiffre de moins peut être intéressant.

En double taille (utilise la ligne en-dessus, donc lignes 0 exclue.

Big(); (ou BigCar) BigBin8(); BigHex8(); BigHex16(); BigDec8();BigDec16(); BigDec9999();

Les noms ne vous plaisent pas? Vous pouvez leur donner un synonyme avec le #define, par exemple

```
#define BD4(x) BigDec9999(x)    // attention, pas de ; final
```

2.2 Afficher des textes

Les caractères ASCII de 0x20 à 0x7F sont connus, mais pas \n et \r.
On passe à la ligne en repositionnant le pointeur avec LiCol();

2.3 Afficher des points et des graphiques

L'origine des coordonnées est en haut à gauche (section 4).
Utiliser l'origine "scolaire" nécessite une simple soustraction: x= (63-x).
Ecrire un point efface les autres pixels du byte écran qu'il contient. Cela n'est pas gênant pour la seule application qui nous intéresse: visualiser des suites de mesures.
Les primitives sont

Dot(x,y); DDot(x,y); Vline(x); Hline(y);

DDot (x,y) affiche 2 points superposés. C'est pratique si on veut visualiser 2 courbes.

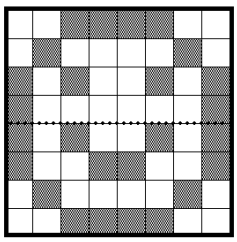
Exercices: Dessiner des diagonales, des segments horizontaux et verticaux, etc
Ecrire des fonctions qui dessinent des segments.

3 Afficher des lutins (sprite)

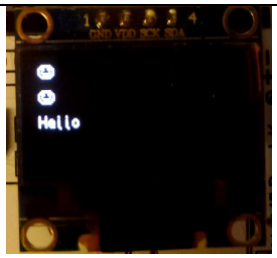
L'écran est fait de bytes alignés verticalement. Licol (li,co); positionne le pointeur sur un byte. Facile de copier une table à partir de cette position.
Pour le lutin smile, la table est

```
  byte  mysmile[]= {0x3c,0x42,0x95,0xa5,0xa1,0xa1, 0xa5,0x95,0x42,0x3c};
```

```
.
void loop(){
  LiCol(0,0); Sprite(smile);
  LiCol(2,0); MySprite(mysmile);
  LiCol(4,0); Text("Hello");
  delay(200);
}
```



3c 42 95 a1 a1 95 42 3c



Lignes obliques, cercles

Ces fonctions ont été écrites et documentées sous https://github.com/nicoud/Oled
La librairie Neopixel offre de plus des remplissages de formes qui ne présentent que peu d'intérêt pédagogique, et conduisent à des programmes gros et lents.

Le grand problème des jeux est la vitesse et la nécessité de déplacer des objets, avec la contrainte que l'on ne peut pas relire l'écran. Neopixel et OledMap construisent en mémoire et copient dans l'écran, ce qui est lent. OledPix dessine directement là où il faut, donc c'est rapide, mais cela écrase parfois plus que ce qu'il faudrait. Un compromis difficile à trouver est donc nécessaire.

jdn 171215(180204