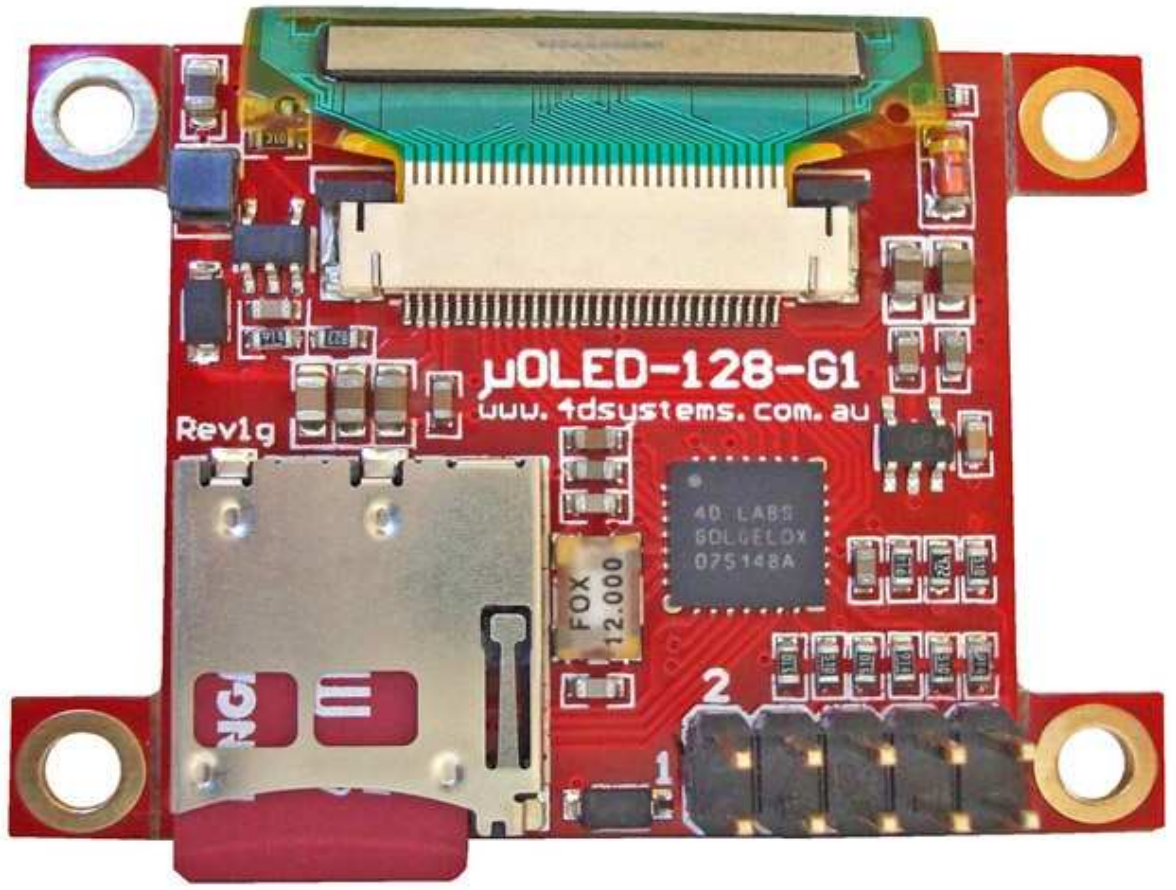
Mise en œuvre d'un afficheur uOLED, exemples avec un PIC18.



L'afficheur utilisé ici est un uOLED-128-G1(SGC), qui peut être commandé un jeu d'instructions simples transmises sur son port série asynchrone.
La version uOLED-128-G1(GFX) de cet afficheur qui possède un puissant logiciel graphique (4DGL) ne sera pas étudiée ici.

La documentation ainsi que tous les utilitaires décrits ici sont disponibles sur http://www.4dsystems.com.au/ onglet « products »
La carte uOLED est disponible (entre autres) chez Farnell et Lextronic. Il existe également un KIT de développement.

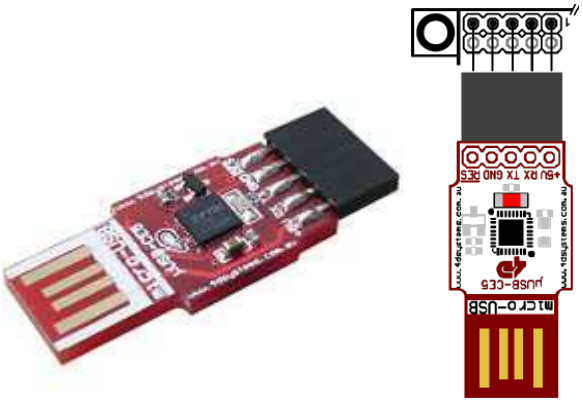
L'afficheur peut être relié au port série asynchrone d'un microcontrôleur, le mode par défaut étant 9600 Bauds, No parity, 1 stop. Cependant l'afficheur dispose d'une fonction « autobaud » lui permettant de se connecter automatiquement à son hôte quelque soit la vitesse de ce dernier.

Le datasheet de l'afficheur décrit entre autre les connexions à réaliser vers un microcontrôleur. (uOLED-128-G1SGC-DS-rev ?.pdf)

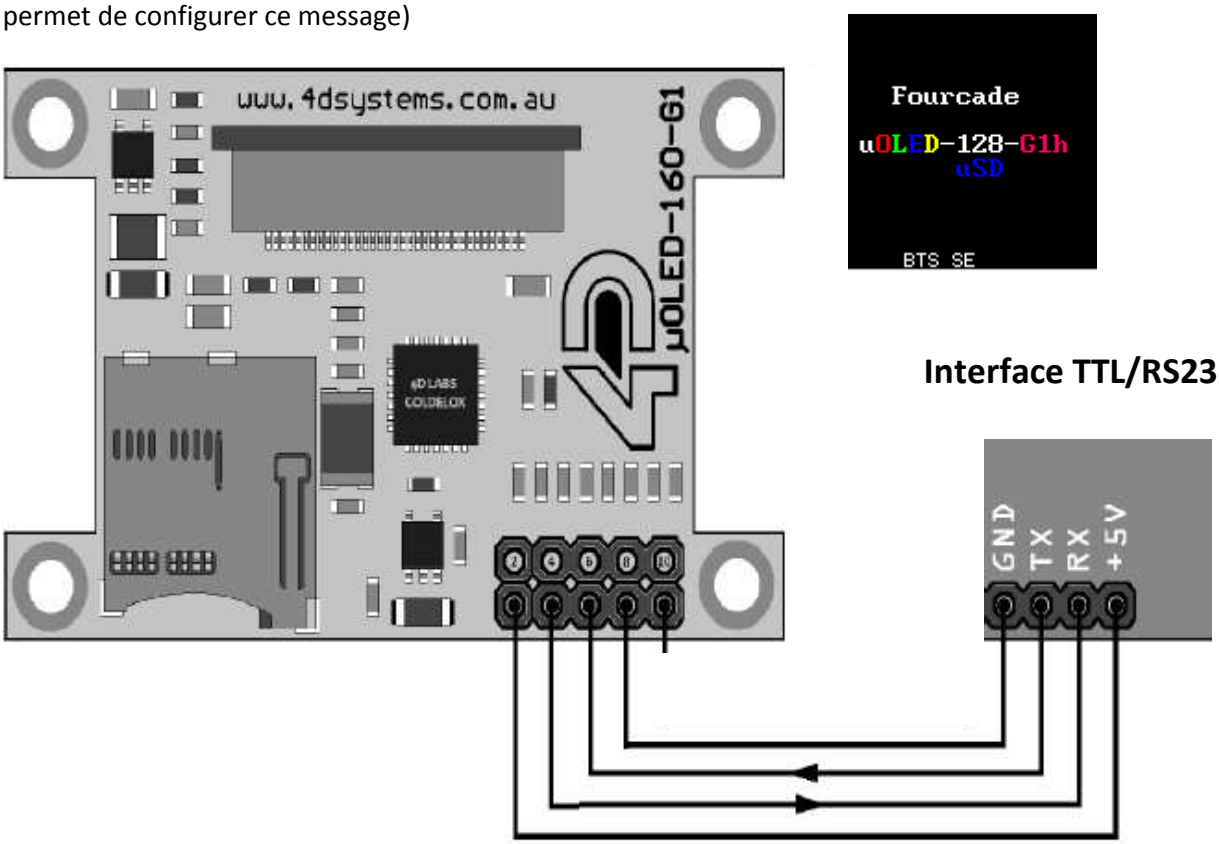
L'afficheur peut également gérer des entrées/sortie. (Boutons, joystick, haut parleur), ces fonctionnalités ne sont pas décrites ici. (De nombreuses videos sur le uOLED-128-G1 sont visualisables sur Internet sur youtube.com par exemple)

1) Test depuis un PC

L'afficheur peut être piloté depuis un PC équipé d'un port série RS232 à l'aide du logiciel FAT-Controller.exe. Ce dernier permet de tester l'ensemble des fonctionnalités de l'afficheur. Il est cependant nécessaire de disposer d'une interface TTL/RS232 pour obtenir une compatibilité électrique avec le port du PC, comme par exemple le uUSB-CE5 de 4DSystems



Après avoir connecté l'afficheur à l'interface TTL/RS232 et ce denier au PC, alimenter ce dernier en 5v, un message de bienvenue, apparait sur l'afficheur. (Le logiciel DISP.EXE permet de configurer ce message)

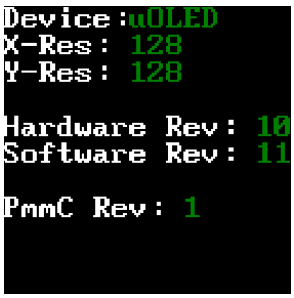


2) Test avec FAT-Controller.exe

Lancer le logiciel FAT-Controller.exe, sélectionner le port COM relié à l'afficheur, la vitesse de 9600 Baud et cliquer « Open ».

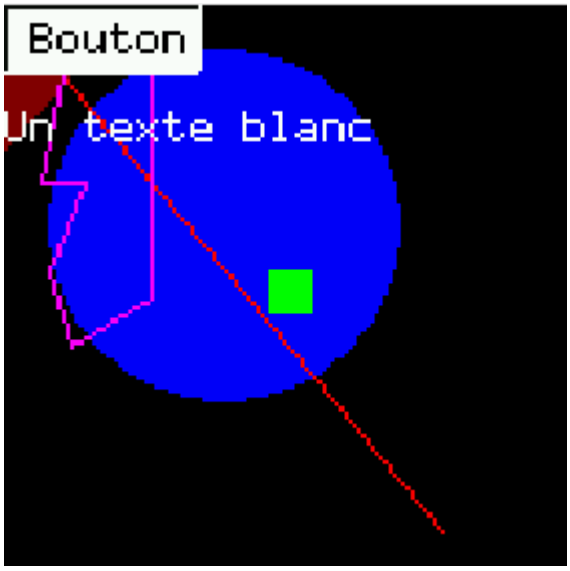
Le logiciel permet de tester l'ensemble des fonctionnalités de l'afficheur, en voici quelque unes :

Onglet « General », la commande « V » (Query Version) permet de visualiser la version de l'afficheur.



La commande « E » (Erase) efface l'afficheur

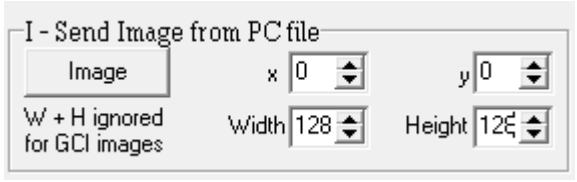
Onglet « Graphic Pt1 » et « Graphic Pt2 » , la commande « C » (Circle) permet de tracer un cercle en indiquant le rayon et les coordonnées du centre, la couleur est également configurable.



Il est facile de tracer des rectangles, triangles, d'écrire un texte etc.

Dans l'onglet « Graphic Pt1 », on peut transférer une image depuis le PC vers l'afficheur. L'image doit avoir au maximum une dimension de 128x128 points. Le logiciel de gestion d'image GIMP (http://www.gimp.org/) permet très simplement de redimensionner une image. (Dans GIMP, onglet « image », puis « échelle et taille de l'image).

Télécharger une image de 128x128 dans l'afficheur,



cliquer le bouton « Image », sélectionner le fichier. L'image apparait (très) lentement sur l'afficheur. Une image de 128x128 pixels de 16bits pèse 32KO, à 9600 Bauds et faut environ 27s pour la transmettre ! Il est possible avec l'onglet « Raw SD » de commander la mémoire flash uSD de 2GO placée sur l'afficheur.
Commande « i » pour initialiser la uSD
Commande « C » pour enregistrer l'image dans la mémoire uSD (utiliser ADh=1 et ADl=0), *soit l'adresse dans la uSD 0x00010000 ou 65536 (décimal)*

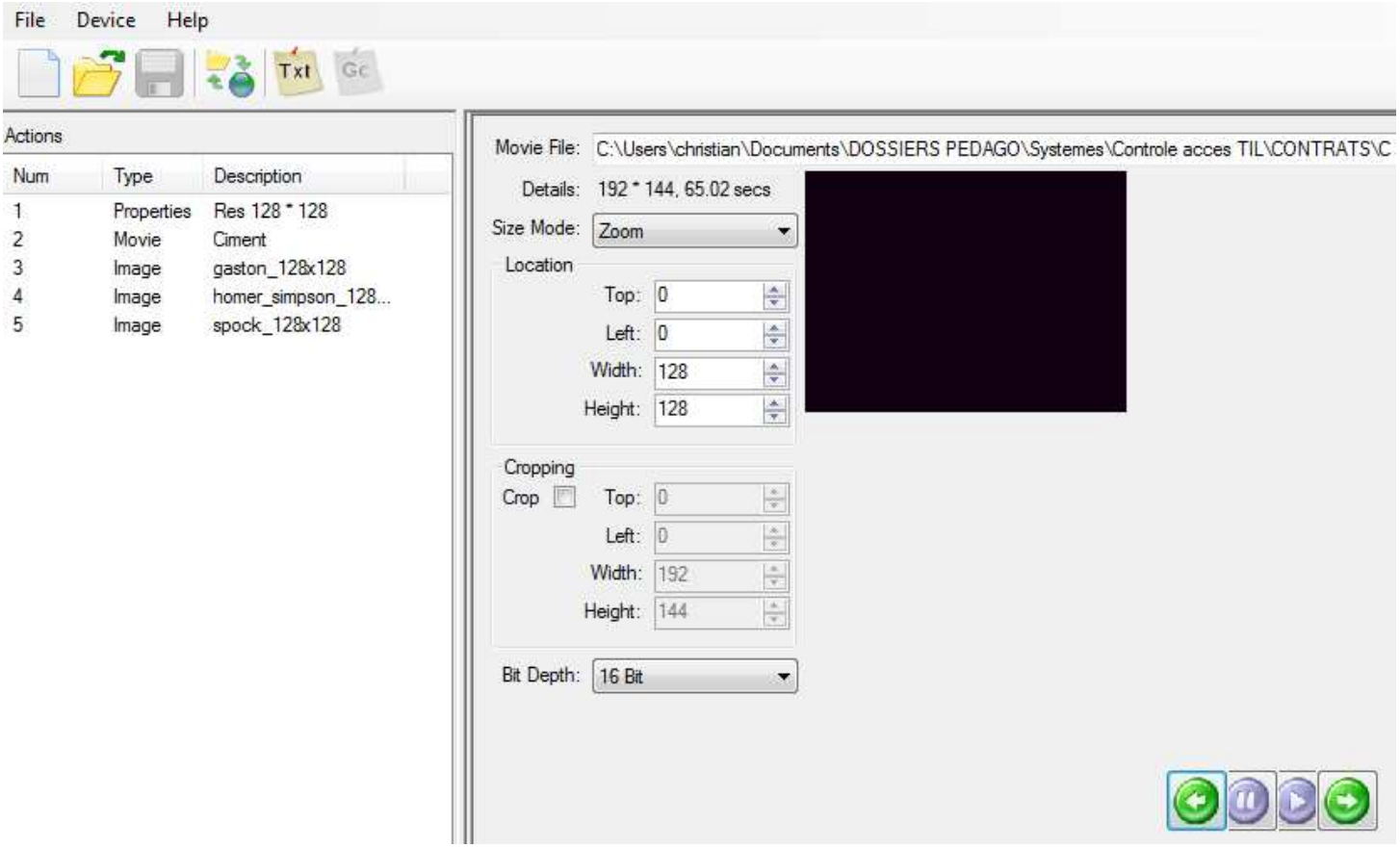
Commande « I » pour recopier une image vers l'écran.
Commande « V » pour jouer une vidéo, il n'y a cependant pas de commande pour recopier une vidéo, il faut donc utiliser un autre logiciel pour enregistrer les médias dans la carte mémoire (GraphicsComposer.exe)

Le PC peut maintenant être remplacé par un microcontrôleur (exemple un PIC18F2620), l'interface TTL/RS232 doit être retiré, les microcontrôleurs étant tolérants TTL (comme l'afficheur). Un simple programme gérant l'USART du microcontrôleur peut piloter l'afficheur. Le jeu d'instruction complet est disponible sur GOLDELOX-SGC-COMMANDS-SIS-rev ?.pdf

3) L'enregistrement de médias avec le logiciel GraphicsComposer.exe

Placer la carte uSD dans un lecteur de carte adapté sur le PC et formater la carte en FAT16.

Lancer « GraphicsComposer.exe » dans la colonne « action », cliquer droit puis ajouter des médias, images et vidéos.



Menu « Devices » , sélectionner « Serial Command Platform » ainsi que le disque uSD. Enregistrer la configuration. Cliquer sur le bouton « Load device » , les données sont transférées (il est possible que le programme demande un formatage, le format étant différent de celui de Microsoft),

Un fichier texte est crée donnant les commandes permettant d'accéder aux médias (attention, en hexadécimal). Ici la commande « `0x40, 0x56, 0x00, 0x10, 0x80, 0x60, 0x10, 0x27, 0x06, 0x5A, 0x00, 0x10, 0x00` » lancera la vidéo.

Organisation d'une carte flash uSD

Les trois derniers nombres représentent l'adresse du media dans la carte uSD.

Un secteur compte 512 Octets.

Une mémoire de 1MO compte 2048 secteurs.

Les addresses de la carte uSD de 2G0 vont de 0 à 1677215 (décimal) soit 0xFFFFFF. Il faut donc trois octets pour coder une adresse.

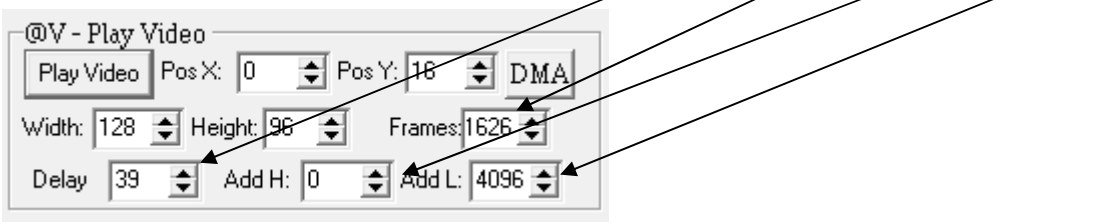
```
============================
SERIAL-PLATFORM OUTPUT FILE
============================
--------------------------------------------------------------------------------
File "Ciment.mpeg"
Sector Address 0x001000
Width = 128 Height = 128 Bits = 16 (Location and size may be adjusted depending on
image size and Size Mode selected)
Display Movie from Memory Card (Serial Command):
Syntax:
@, V, x, y, width, height, colourMode, delay, frames(msb), frames(lsb),
SectorAdd(hi), SectorAdd(mid), SectorAdd(lo)
Data:
0x40, 0x56, 0x00, 0x10, 0x80, 0x60, 0x10, 0x27, 0x06, 0x5A, 0x00, 0x10, 0x00

--------------------------------------------------------------------------------
File "gaston_128x128.jpg"
Sector Address 0x0140E0
Width = 128 Height = 128 Bits = 16 (Location and size may be adjusted depending on
image size and Size Mode selected)
Display Image from Memory Card (Serial Command):
Syntax:
@, I, x, y, width, height, colourMode, SectorAdd(hi), SectorAdd(mid), SectorAdd(lo)
Data:
0x40, 0x49, 0x00, 0x00, 0x80, 0x80, 0x10, 0x01, 0x40, 0xE0

--------------------------------------------------------------------------------
File "homer_simpson_128x128.jpg"
Sector Address 0x014120
Width = 128 Height = 128 Bits = 16 (Location and size may be adjusted depending on
image size and Size Mode selected)
Display Image from Memory Card (Serial Command):
Syntax:
@, I, x, y, width, height, colourMode, SectorAdd(hi), SectorAdd(mid), SectorAdd(lo)
Data:
0x40, 0x49, 0x00, 0x00, 0x80, 0x80, 0x10, 0x01, 0x41, 0x20

--------------------------------------------------------------------------------
File "spock_128x128.jpg"
Sector Address 0x014160
Width = 128 Height = 128 Bits = 16 (Location and size may be adjusted depending on
image size and Size Mode selected)
Display Image from Memory Card (Serial Command):
Syntax:
@, I, x, y, width, height, colourMode, SectorAdd(hi), SectorAdd(mid), SectorAdd(lo)
Data:
0x40, 0x49, 0x00, 0x00, 0x80, 0x80, 0x10, 0x01, 0x41, 0x60

--------------------------------------------------------------------------------
```

Rappel de la commande

Les trois derniers octets indiquent l'adresse absolue dans la uSD (ici 0 et 4096 en décimal)

Après avoir replacé la mémoire sur l'afficheur il est possible avec FAT-Controller.exe de visualiser vidéos et images en utilisant les paramètres donnés ci-dessus (ATTENTION, les paramètres doivent être entrés en décimal).

```
@, V, x, y, width, height, colourMode, delay, frames(msb), frames(lsb),
SectorAdd(hi), SectorAdd(mid), SectorAdd(lo)
Data:
0x40, 0x56, 0x00, 0x10, 0x80, 0x60, 0x10, 0x27, 0x06, 0x5A, 0x00, 0x10, 0x00
```



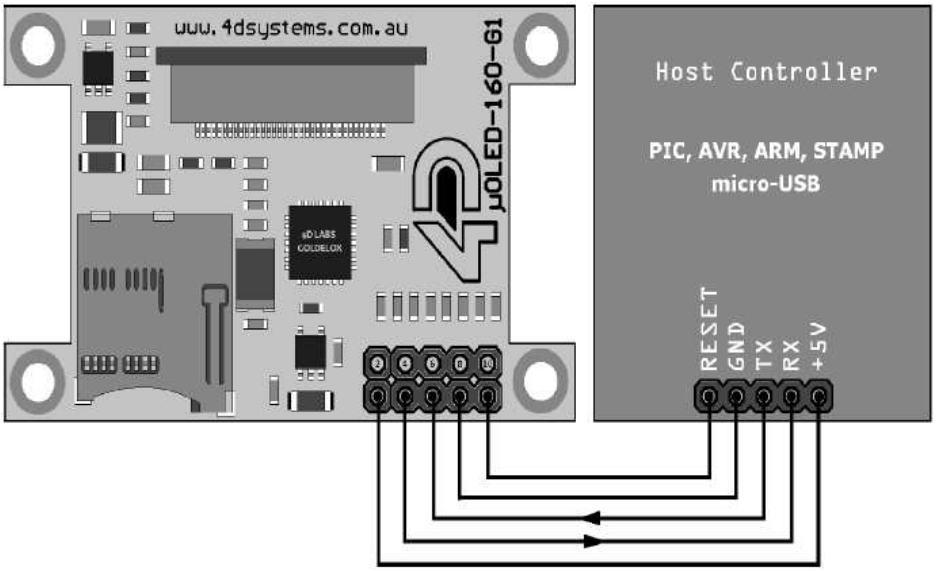
Commencer par Init uSD puis Play Video.

4) Commander l'afficheur avec un PIC18

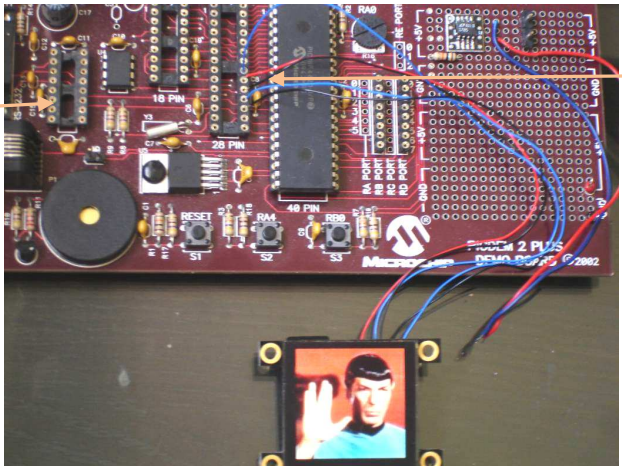
L'afficheur uOLED est connecté directement avec l'USART du PIC18 (croiser RX et TX). Il est possible d'utiliser un P18F2620 fonctionnant sous 5v ou la nouvelle génération P18F26J50 fonctionnant sous 3.3v, l'afficheur ayant une sortie 3.3v 50mA.

On peut utiliser une carte PICDEM2+ à condition de retirer le circuit d'interface TTL/RS232 MAX232.

Le RESET de l'afficheur peut être relié à n'importe quel port du PIC.



Interface RS232 retiré



Le module uOLED est connecté sur le connecteur DIL28

L'exemple de programme de gestion du module uOLED est destiné à une carte PICDEM2+ avec microcontrôleur P18F4620. L'interface RS232 DOIT être retiré.

Le module uOLED est sur le connecteur 28 broches du PÏCDEM2+ :
Broche 17 (TX) reliée à RX du uOLED
Broche 18 (RX) reliée à TX du uOLED
Broche 19 (0v) reliée à GND du uOLED
Broche 20 (VDD) reliée à +5v du uOLED
Broche 21 (PB0) reliée à RESET du uOLED

L'exemple peut être très facilement adapté à n'importe quel PIC18. En l'absence d'afficheur LCD, tous les appels aux fonctions de la bibliothèque xlcd seront retirés.

Le PIC18F4620 est en mode horloge interne 8MHz (fichier init_horloge_interne.c), aucun quartz ou oscillateur externe n'est donc nécessaire.

Le programme montre la mise en œuvre de quelques fonctions graphiques :
Affichages de textes, d'éléments graphiques simples (rectangles, cercles), d'images et d'une vidéo. Ces deux derniers éléments doivent avoir été au préalable chargés en uSD comme décrit précédemment. Les adresses en uSD des objets sont recopiés dans le le programme pour le PIC18

```
// C.Dupaty 09/2010
// test_uOLED.c
// demo gestion afficheur graphique uOLED-128-G1 (SGC)
// les équivalences (#define) des commandes sont dans GSGCdef.h
// le descritif des commande uOLED est dans : GOLDELOX-SGC-COMMANDS-SIS-rev3.pdf
// le description des connexions et utilitaires uOLED sont dans : uOLED-128-G1SGC-DS-rev4.pdf
// La carte uSD a été chargé depuis un PC avec 3 images et une vidéo à partir de l'utilitaire
GraphicsComposer.exe
// ce dernier a genere un fichier test.txt qui contient entre autre les adresses de ce objets
dans la mémoire uSD
// IMPORTANT !!! le module uOLED est connecté à une carte PICDEM2+ SANS l'interface RS232.
//
// Microcontroleur P18F4620 sur horloge interne 8MHz, le module uOLED est connecté sur
// le connecteur DIL 28 broches,
// 17 : TX (donc RX de uOLED)
// 18 : RX (donc TX de uOLED)
// 19 : 0v
// 20 : 5v
// 21 : Reset uOLED
//
// Ce programme peut être utilise sans afficheur LCD ni PICDEM2+ (retirer alors tous les
// appels aux fonctions de gestion du LCD)

#include <p18f4620.h>
#include <stdio.h>  // pour fprintf
#include "xlcd.h"     // lib cree avec MAESTRO pour l'afficheur LCD (ici PICDEM2+)
#include "GSGCdef.h"  // commandes uOLED

#define S2 PORTAbits.RA4             // bouton S2 sur PICDEM2+
#define reset PORTBbits.RB0
#define TRISreset TRISBbits.TRISB0

#define SPOCK  0x014160       // adresses des images dans la uSD (voir fichier test.txt)
#define GASTON        0x0140E0
#define OMER   0x014120
#define BINGO  0x001000       // adress de la video

unsigned char chaine1[]="Test uOLED-128-G1 (SGC)";

void putsci(unsigned char c)
{
       while(!TXSTAbits.TRMT);
    TXREG=c;                          /* envoie un caractère */
    while(!PIR1bits.TXIF);
}

char data_recue(void)                // reception d'unedonnee
{
       if (PIR1bits.RCIF)            /* char recu en reception*/
       {
              PIR1bits.RCIF=0;       // efface drapeau
              return (1);    // indique qu'un nouveau caractère est dans RCREG
       }
       else return (0);              // pas de nouveau caractère reçu
}

char ack(void)
{
              while (!data_recue());
              return(RCREG);
}

void rectangle(unsigned char x1,unsigned char y1,unsigned char x2,unsigned char y2,unsigned
int color)
{
       putsci(GSGC_RECTANGLE);
       putsci(x1);
       putsci(y1);
       putsci(x2);
       putsci(y2);
```

```
	putsci(color>>8);
	putsci(color&0x00FF);
}

void cercle(unsigned char x,unsigned char y,unsigned char radius,unsigned int color)
{
	putsci(GSGC_CIRCLE);
	putsci(x);
	putsci(y);
	putsci(radius);
	putsci(color>>8);
	putsci(color&0x00FF);
}

void afftexte_graphic(char x, char y, char font,int color , char w, char h, char * mess)
{
	putsci(GSGC_STRINGGFX);
	putsci(x);
	putsci(y);
	putsci(font);
	putsci(color>>8);
	putsci(color&0x00FF);
	putsci(w);
	putsci(h);
	fprintf(_H_USART,"texte NON formate %s",mess);
	putsci(0);
}

void afftexte_text(char x,char y,char font,int color,char* mess)
{
	putsci(GSGC_STRINGTXT);
	putsci(x);
	putsci(y);
	putsci(font);
	putsci(color>>8);
	putsci(color&0x00FF);
	fprintf(_H_USART,"texte formate : %s",mess);
	putsci(0);
}

void affiche_image(char x, char y, char w, char h, char color, unsigned long ad)
{
	putsci('@');
	putsci(GSGC_MCIMAGE);
	putsci(x);
	putsci(y);
	putsci(w);
	putsci(h);
	putsci(color);
	putsci(ad>>16);
	putsci((ad>>8)&0x0000FF);
	putsci(ad&0x0000FF);
}

void affiche_video(char x, char y, char w, char h, char color, char delay, int frames,unsigned
long ad)
{
	putsci('@');
	putsci(GSGC_MCVIDEO);
	putsci(x);
	putsci(y);
	putsci(w);
	putsci(h);
	putsci(color);
	putsci(delay);
	putsci(frames>>8);
	putsci(frames&0x00FF);
	putsci(ad>>16);
	putsci((ad>>8)&0x0000FF);
	putsci(ad&0x0000FF);
```

```
}

void efface(void)
{unsigned int i;
               putsci(GSGC_CLS);            // efface ecran
               for(i=0;i<20000;i++);
}

  // dirige user_putc vers l'afficheur LCD du PD2+
int _user_putc (char c)
{
       XLCDPut(c);
}


void main(void)
{
unsigned long i;

       init_horloge_interne();
       TRISCbits.TRISC7 = 1;  // RX - Set Recieve pin for tristate
       TRISCbits.TRISC6 = 0;  // TX - Set as Output
       TRISreset = 0;        // RESET uOLED en sortie

       SPBRGH= 0x06;                                        // SPBRG est compose de SPBRGH et
SPBRG
       SPBRG = 51;                                  /* configure la vitesse (BAUD) 9600 N 8
1*/
    TXSTA = 0b00100000;
    RCSTA = 0b10010000;                                             /* active l'USART*/
       BAUDCONbits.BRG16=0;  // SPBRGH = 0 pour 9600 (ignore SPBRGH et 1 pour 1200
       TXSTAbits.BRGH=0;            // High Speed

       XLCDInit();
       XLCDL1home();                                //ligne 0 de l'afficheur
       fprintf(_H_USER, "TEST uOLED       " );     // vers LCD
       reset=1;                                             // Reset module uOLED
       reset=0;
       Nop(); Nop(); Nop(); Nop();
       reset=1;
       do
       {
              putsci(GSGC_AUTOBAUD);       // test connexion uOLED
              while (!data_recue());
       }
       while (RCREG !=ACK);                 // si ACK=1 les communications sont établies
       while(1)
       {
              efface();
              XLCDL2home();
              fprintf(_H_USER, "OK ! pressez S2 ");
              while(S2);
              while(!S2);

              putsci(GSGC_VERSION); // affiche infos uOLED
              putsci(0x01);
              XLCDL2home();
              fprintf(_H_USER, "S2: infos        ");
              while(S2);
              while(!S2);

              efface();
              rectangle(10,10,100,80,RED); // affiche un rectangle rouge plein
              XLCDL2home();
              fprintf(_H_USER, "S2: rectangle    ");
              while(S2);
              while(!S2);

              cercle(50,90,30,BLUE); // affiche un cercle bleu plein
              XLCDL2home();
```

```
                fprintf(_H_USER, "S2: cercle    ");
                while(S2);
                while(!S2);

                efface();
                afftexte_text(2,2,FONT3,RED,chaine1); // affiche un texte formate
                XLCDL2home();
                fprintf(_H_USER, "texte FORMATE    ");
                while(S2);
                while(!S2);

                efface();
                afftexte_graphic(10,12,FONT2,GREEN,2,2,chaine1); // affiche un texte graphique
                XLCDL2home();
                fprintf(_H_USER, "texte GRAPHIC ");
                while(S2);
                while(!S2);

                efface();
                affiche_image(0,0,0x80,0x80,COLOR16,SPOCK); // SPOCK
                XLCDL2home();
                fprintf(_H_USER, "S2: Spock        ");
                while(S2);
                while(!S2);

                affiche_image(0,0,0x80,0x80,COLOR16,GASTON); // GASTON
                XLCDL2home();
                fprintf(_H_USER, "S2: Gaston        ");
                while(S2);
                while(!S2);

                affiche_image(0,0,0x80,0x80,COLOR16,OMER); // OMER
                XLCDL2home();
                fprintf(_H_USER, "S2: Omer          ");
                while(S2);
                while(!S2);
                //
                efface();
                XLCDL2home();
                fprintf(_H_USER, "Video BINGO     ");       // une video
                affiche_video(0,0x10,0x80,0x60,COLOR16,0x27,0x065A,BINGO);
                while(ack()!=ACK);
                XLCDL2home();
                fprintf(_H_USER, "recommencer : S2");       // une video
                while(S2);
                while(!S2);
        }
}
```