

CHAPITRE 4

Développement et réalisation du projet

4.1INTRODUCTION

- ✓ Pour la réalisation de notre projet nous avons eu recours à l'utilisation de plusieurs composants dont les principaux sont :
 - Le PIC32MX795F512H pour le développement du programme.
 - L'encrant SHARP LQ043T3DX05 pour afficher le contenu de USB
 - Mémoire SRAM ISSI pour mémoriser l'image.
 - Connecteur USB femelle.

- ✓ Pour la programmation nous avons utilisé :
 - MBLAB IDE
 - MBLAB C32
 - Microchip USB OTG Configuration Tool.
 - Microchip FS stack.
 - Microchip Graphics Stack.

4.2 Réalisation de la carte électronique :

4.2.1 Ecran SHARP LQ043T3DX05(PSP 1000) :

Cet écran est une matrice colorée à cristaux liquides TFT (le Transistor de Film Mince) **sans contrôleur** ; Il est composé :

- d'un panneau TFT-À-CRISTAUX-LIQUIDES coloré,
- le pilote ICs,
- Une entrée FPC
- une unité de rétro éclairage (back light).

Le Graphisme et textes peuvent être affichés sur le panneau 480 * 3 * 272 points de 16 millions de couleurs, en fournissant des signaux de données 24bit (8bit * RGB).

Cet écran est géré par quatre signaux de synchronisation (**HSYNC**, **VSYNC**, **PCKL** et **le DEN**) et un signal pour le rétro éclairage. Les tensions logiques respectives à ces signaux sont en (type +2.5v) et le dernier en (type +5v).



Figure 4.1 l'écran SHARP LQ043T3DX05

N°	Nom	Fonction	N°	Nom	Fonction
1	GND	GND(0V)	21	B0	Donnée du Signal bleu(LSB)
2	GND	GND(0V)	22	B1	Donnée du Signal bleu
3	VCC	+2.5V	23	B2	Donnée du Signal bleu
4	VCC	+2.5V	24	B3	Donnée du Signal bleu
5	R0	Donnée du Signal rouge (LSB)	25	B4	Donnée du Signal bleu
6	R1	Donnée du Signal rouge	26	B5	Donnée du Signal bleu
7	R2	Donnée du Signal rouge	27	B6	Donnée du Signal bleu
8	R3	Donnée du Signal rouge	28	B7	Donnée du Signal bleu (MBS)
9	R4	Donnée du Signal rouge	29	GND	GND(0V)
10	R5	Donnée du Signal rouge	30	CK	Signal de l'horloge
11	R6	Donnée du Signal rouge	31	DISP	Signal ON\OFF
12	R7	Donnée du Signal rouge (MSB)	32	Hsync	Synchronisation horizontale
13	G0	Donnée du Signal vert(LSB)	33	Vsync	Synchronisation verticale
14	G1	Donnée du Signal vert	34	NC	NC
15	G2	Donnée du Signal vert	35	AVDD	+5V
16	G3	Donnée du Signal vert	36	AVDD	+5V
17	G4	Donnée du Signal vert	37	NC	NC
18	G5	Donnée du Signal vert	38	TEST1	+5V
19	G6	Donnée du Signal vert	39	TEST2	GND
20	G7	Donnée du Signal vert(MSB)	40	TEST3	GND

Tableau 4.1 description de signaux

4.2.2 Mémoire RAM ISSI 32Ko*8 ISSIIS61C256AH

4.2.2.1 caractéristiques :

- Fonctionnement entièrement statique : aucune horloge ni régénération est exigée.
- Son alimentation est en 5V.

4.2.2.2 description :

Cette mémoire SRAM fonctionne en très grande vitesse en basse puissance, sa taille est de 32,768 mot de RAMs statiques de 8 bit. Son temps d'accès est de 10ns. Elle est fabriquée en utilisant la technologie de CMOS.

Dans notre projet, nous avons utilisé 4 composants de cette mémoire pour créer le frame buffer de notre Ecran. Nous avons ajouté les signaux CE de chaque mémoire au le bus d'adresse pour pouvoir balayer toutes les cases de la mémoire globale.

Le boîtier de la mémoire statique utilisée dans le projet est donné dans la figure 4.2).

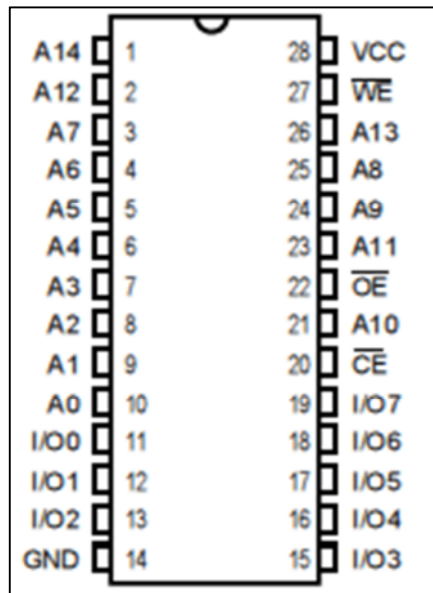
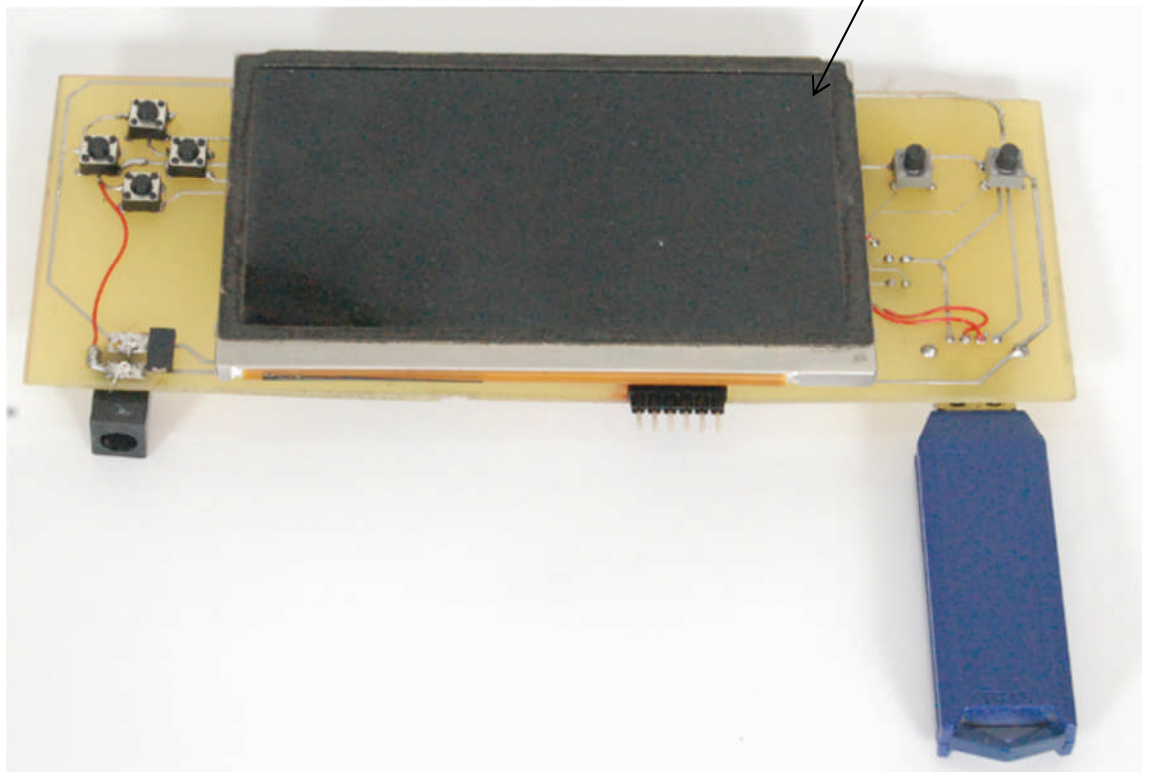
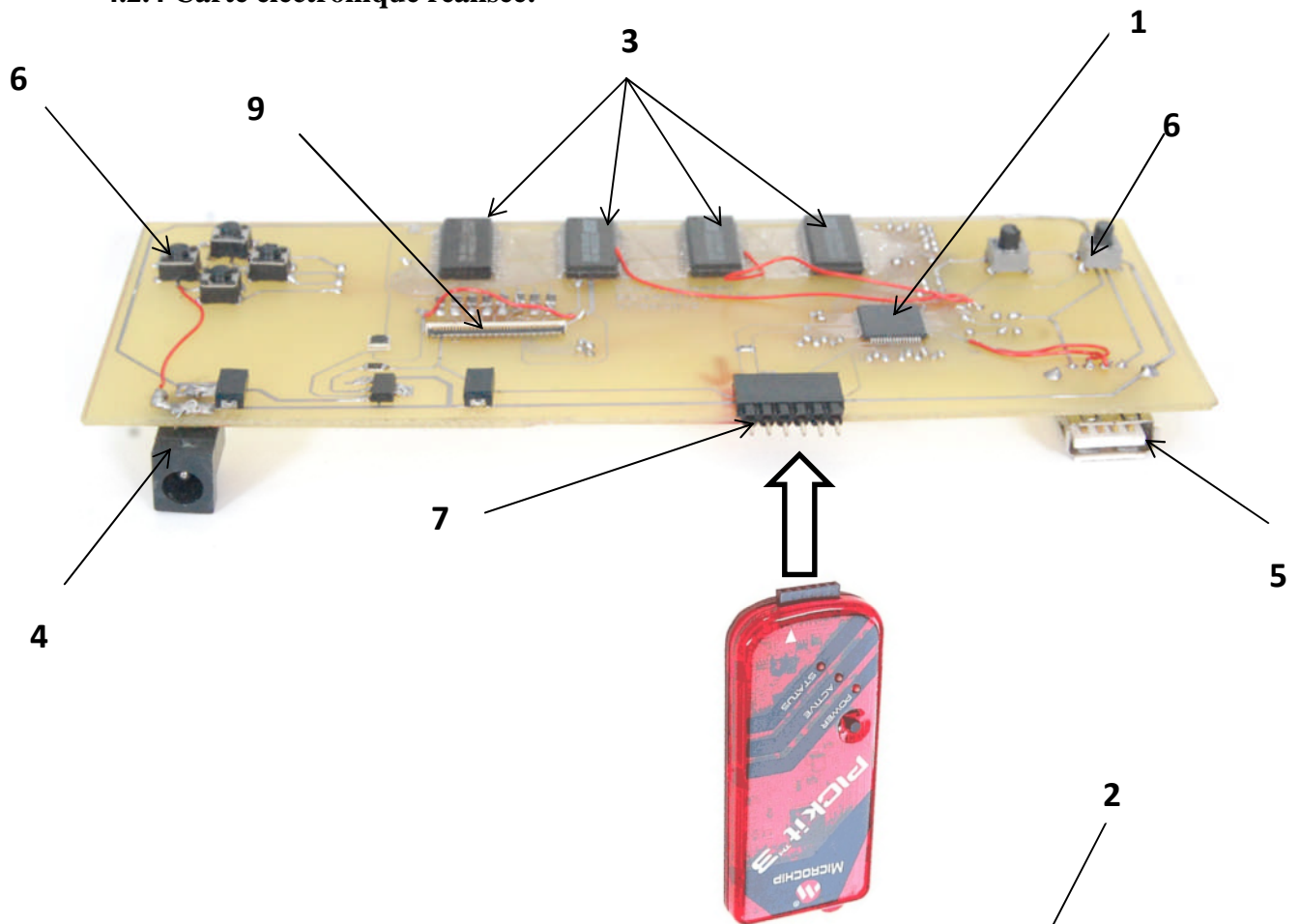


Figure 4.2 Brochage de la mémoire SRAM de ISSI

4.2.3 Constitution de la carte :

1. le PIC32MX795F512H
2. Ecran SHARP LQ043T3DX05(PSP 1000) .
3. SRAM connecté au PMP et l'écran LCD .
4. Connecteur d'alimentation externe.
5. Connecteur USB femelle.
6. Boutons poussoirs.
7. Connecteur PICKIT3
8. Oscillateur à quartz à 8MHz. Cet oscillateur est utilisé pour fournir une horloge à 80MHz via la PLL interne du PIC32MX.
9. Connecteur FH12A-40S-0.5SH pour brancher l'écran TFT.
10. Régulateurs qui fournissent les tensions de 2.5v et 5v à la carte.

4.2.4 Carte électronique réalisée:



4.3 développement du programme :

4.3.1 le contrôleur graphique(LCC) :

En général, le contrôleur graphique envoie les informations des pixels à Ecran d'affichage de manière continue à un certain taux. En général la fréquence de rafraichissement tourne autour de 60 Hz.

À la première inspection, il semble que cette tâche prenne la plupart du temps CPU du microcontrôleur.

Cependant, ce n'est pas le cas pour PIC32MX 795F512H qui contient un module DMA permettant un transfert de données sans l'intervention de processeur. En exploitant le module DMA pour le transfert du Stream video , moins de 5 % de temps CPU est consommée pour cette tache. Cette fonction est appelée : contrôleur graphique "virtuel".

Le périphérique DMA du PIC32MX795F512H peut transférer des données d'un emplacement à un autre sans intervention CPU (voir annexe page 56 fonctionnement de DMA figure 1.3).

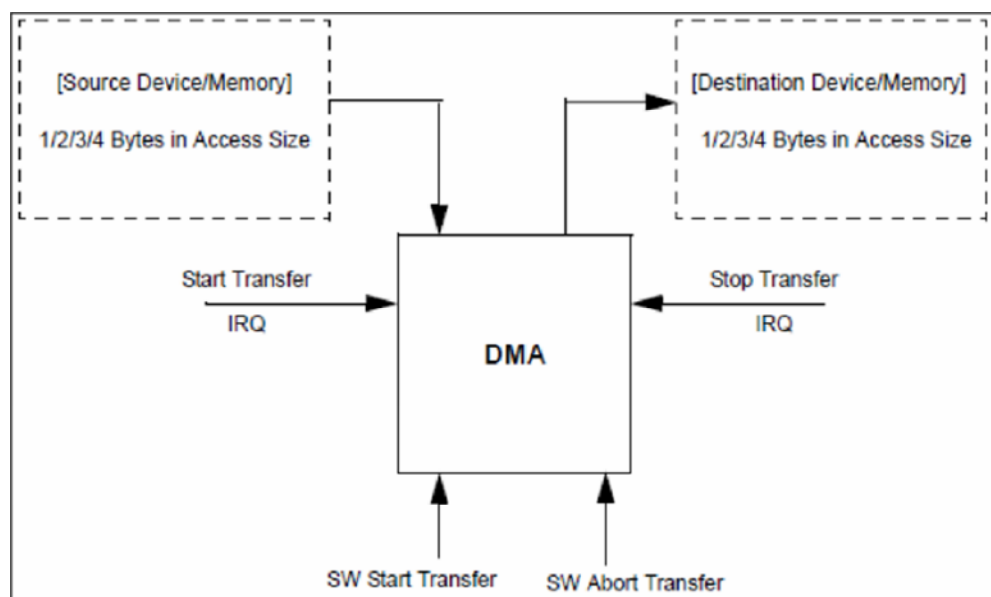


Figure 4.3 : Fonctionnement du DMA

Dans contrôleur graphique virtuel, la DMA est configurée pour transférer le contenu d'une **ligne de données** par le Port de Maître Parallèle(PMP)(voir annexe page 60).

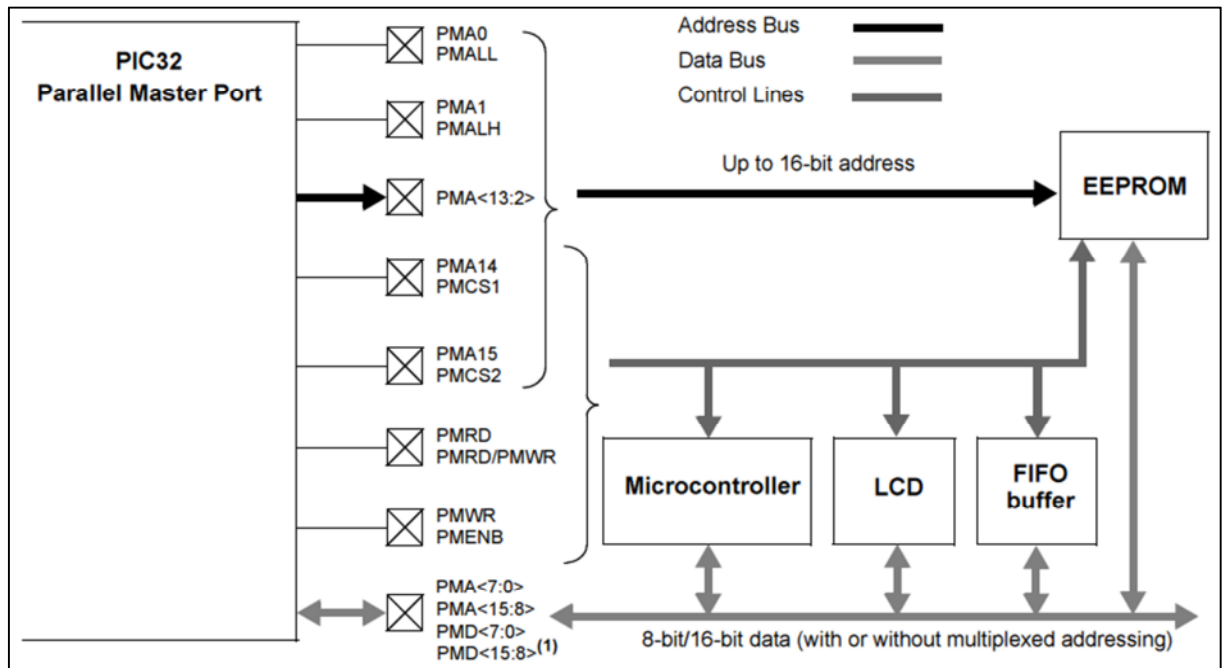


Figure 4.4 Fonctionnement du PMP

Chaque ligne est constituée d'un nombre important de pixels. La DMA envoie une partie de la mémoire pendant un transfert.

Un PMP (ou un Timer) interrompt la demande, et, est utilisé pour déclencher le transfert de DMA suivant, jusqu'à ce qu'une ligne soit transférée.

Dans les dispositifs PIC32, un Timer est utilisé calibrer le débit de transfert des données par la DMA.

Pendant le transfert de données, le signal stropsePMP(lu ou écrit) après chaque transfert de pixel.

Le read/writestropse du périphérique PMP agit comme l'horloge de pixel pour l'Ecran d'affichage.

Après que chaque ligne de données de pixel est transférée, le CPU est interrompu par la DMA ; Certains signaux de commande (par exemple : HSYNC, VSYNC et le DEN) nécessaires pour l'afficheur LCD sont mis à jour.

Ceci est répété continuellement jusqu'à ce que toute l'image de l'écran est transférée depuis le frame buffer (une mémoire RAM) vers l'Ecran.

Dans cette configuration, la mémoire SRAM externe est utilisée ; Cette configuration est la base pour un système graphique sans contrôleur.

Le système peut être installé pour utiliser la mémoire SRAM interne ou la mémoire SRAM externe ; On peut voir un diagramme de chaque système dans la Figure 4.5 et La figure4.6.

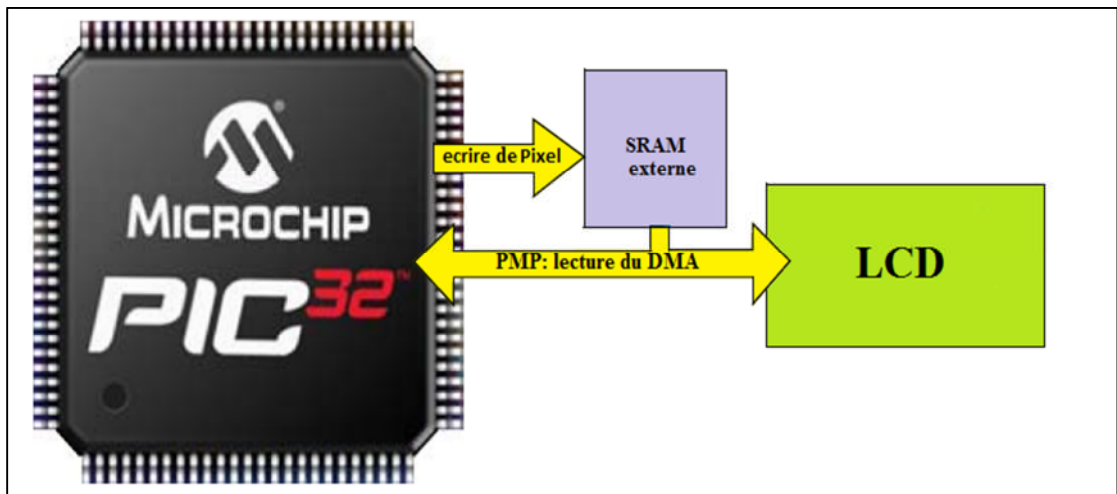


figure 4.5 mémoire externe

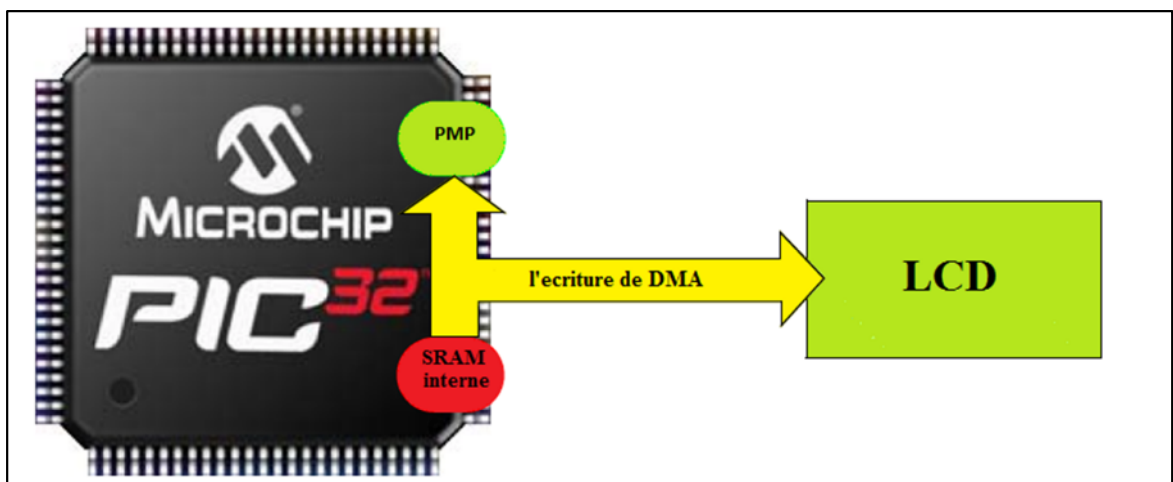


Figure 4.6 mémoire interne

Les signaux HSYNC, VSYNC, le DEN et PCLK sont tous utilisés pour synchroniser les données de pixel avec le cadre graphique et le panneau LCD.

Les lignes syncindique au panneauLCD quand les données sont au début ou à la fin d'une ligne (**HSYNC**) ou un cadre (**VSYNC**).

Le signal DEN active l'écran pur recevoir des données de l'extérieur.

Nous nous sommes servi du programme LCC développé par la société Microchip pour réaliser le contrôle virtuel de l'écran, donc nous l'avons paramétrer à notre carte en terme :

- La largeur d'impulsion,
- Les broches des signaux de synchronisation V,H,CLK.

Une fois que ces valeurs sont ajoutés dans le pilote LCC, le Panneau à cristaux liquides montre (affiche) le cadre. Les paramètres du LCD SHARP LQ043T3DX05 sont donnés dans le tableau suivant :

Parameter		Symbol	Min.	Typ.	Max.	Unit	Remark
Clock	Frequency	1/Tc	7.83	9.00	9.26	MHz	
	Duty ratio	Th/T	40	50	60	%	
Data	Set up time	Tds	25	—	—	ns	
	Hold time	Tdh	25	—	—	ns	
Horizontal synchronizing HSYNC	Period	TH	—	525	—	Clock	
	Pulse width	THp	—	41	—	Clock	
	Horizontal period	THd	—	480	—	Clock	
	Back porch	THb	—	2	—	Clock	
	Front porch	THf	—	2	—	Clock	
Vertical synchronizing VSYNC	Period	TV	—	286	—	Line	
	Pulse width	TVp	—	10	—	Line	
	Vertical period	TVd	—	272	—	Line	
	Back porch	TVb	—	2	—	Line	
	Front porch	TVf	—	2	—	Line	

Tableau 1.4 les valeurs des constantes de l'écran LCD SHARP LQ043T3DX05

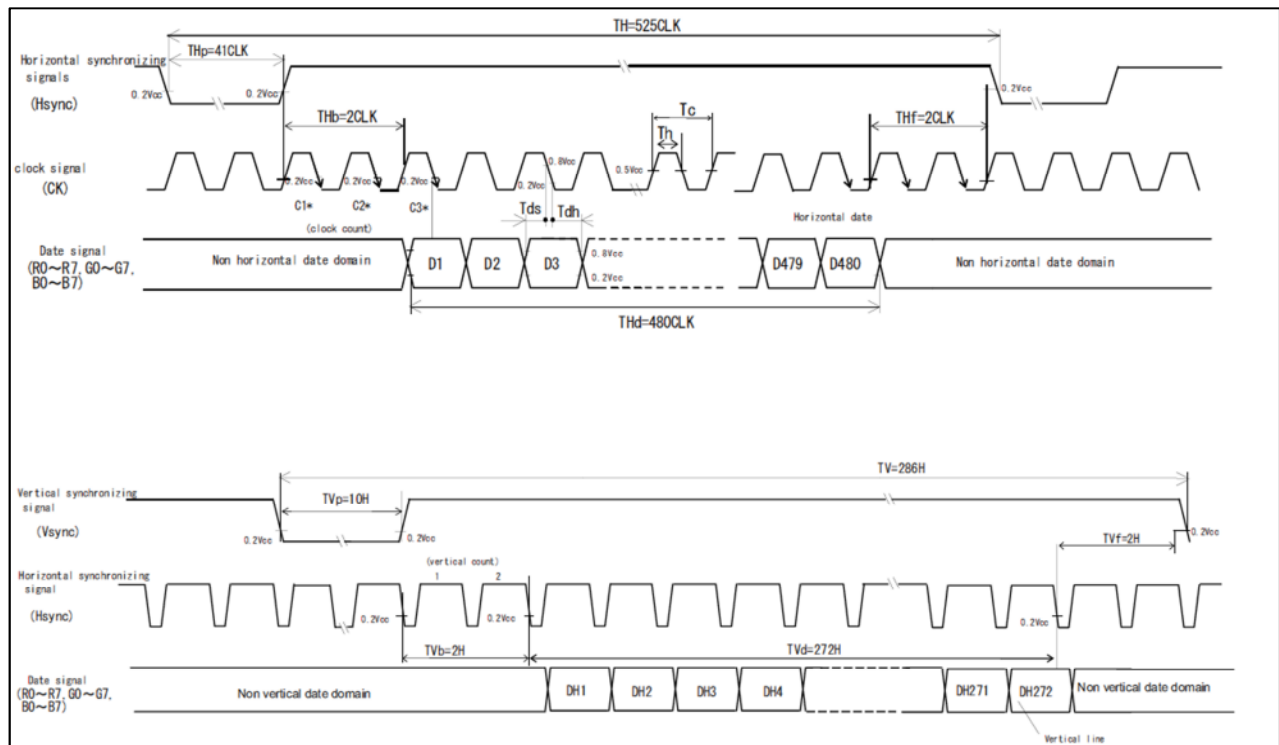


Figure 4.7 les temps des signaux d'entrée

- ORGANIGRAMME DE LCC :

La figure 4.8 montre un organigramme expliquant le fonctionnement du contrôleur graphique virtuel à l'intérieur du microcontrôleur PIC32.

Le bloc intitulé DMA/PMP nous montre les rôles des modules DMA et PMP qui partagent avec le CPU le même bus de données.

Le bloc intitulé CPU montre les tâches nécessaires pour le rendu graphique.

Le DMA ISR est le **seul code** qui doit **être écrit** après l'activation et l'initialisation de la DMA et le PMP pour envoyer le contenu du framebuffer à un afficheur.

Cet organigramme ne spécifie pas où le cadre graphique est stocké (intérieurement ou extérieurement) il ne décrit pas non plus de mise à jour de l'image de cadre.

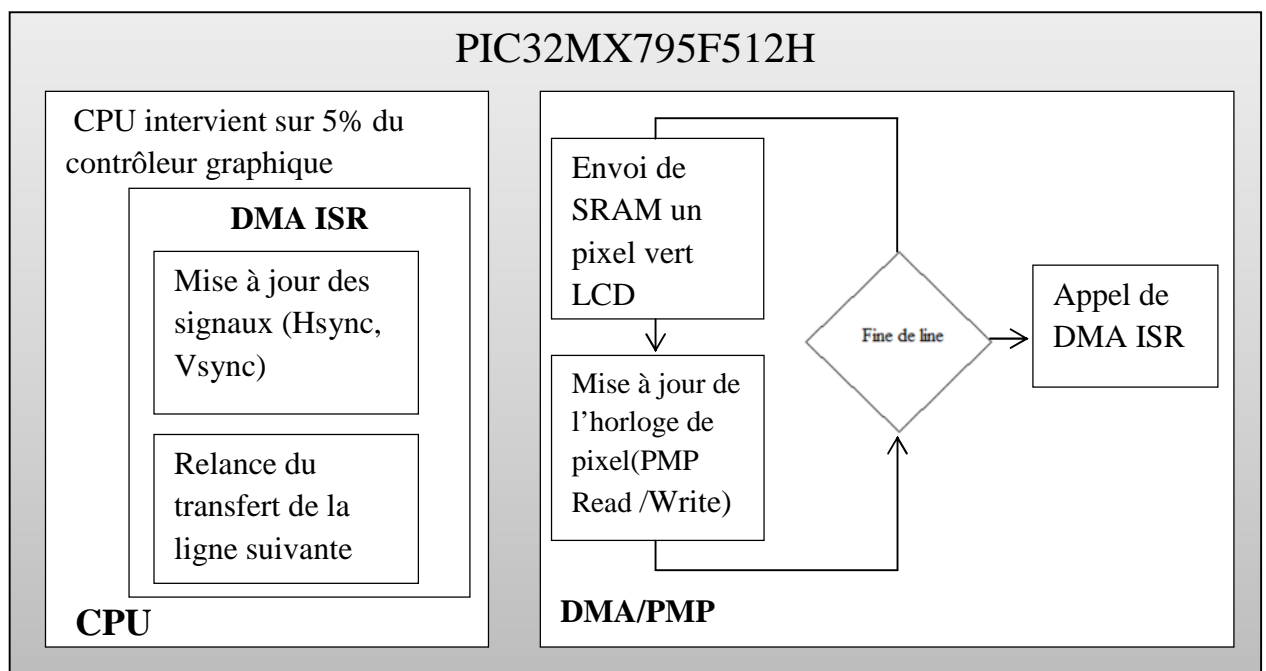


Figure4.8 fonctionnement du module LCC dans le PIC32MX

- **RENDU DE NOUVEAUX PIXELS SUR LE frame buffer Putpixel() :**

Le rendu de nouveaux pixels dans l'écran est aussi important que le rafraîchissement de l'écran. Cette tâche est exécutée par le CPU.

Si le cadre est stocké extérieurement, le transfert de **DMA est suspendu** tandis que le cadre est mis à jour. Ceci est nécessaire car il y a seulement un périphérique PMP et il est partagé entre l'écran et la mémoire SRAM externe (voir Figure 4.9 ci-dessous).

Cette méthode **affecte** vraiment le **taux** de rafraîchissement l'écran.

La quantité de mises à jour de pixel doit être contrôlée pour empêcher un trop grand changement du taux de rafraîchissement, autrement, le changement sera perceptible par l'œil humaine. Ceci est fait en utilisant un **compteur de pixel** dans le contrôleur graphique "virtuel" qui est mis à jour sur chaque pixel écrit et dégagé pendant chaque interruption DMA.

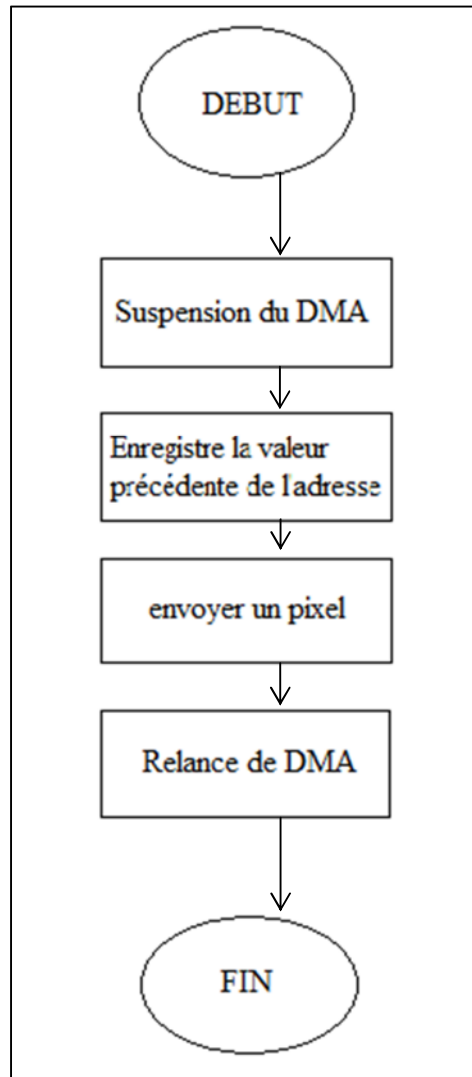


Figure 4.9 Organigramme de fonctionnement de `Putpixel()` .

4.3.2 configuration de l'USB OTG :

Le PIC32MX795F512H intègre un module USB pouvant fonctionner en mode esclave, maître et OTG.

Ce module réalise les fonctions hardware nécessaires à l'USB, mais pour que le μ C puisse communiquer par USB, le code doit gérer tout le protocole USB.

Cette gestion exige une connaissance approfondie sur le protocole USB. La société Microchip a fourni donc un framework configurable.

En plus de ce framework, Microchip fournit des bibliothèques correspondant à différents drivers de classes.

Dans ce projet, le μ C sera configuré en mode USB HOST(hôte) et nous utiliserons le framework USB de Microchip ainsi que la bibliothèque de la classe MSD (Masse Storage Device Class)(voir Chapitre1 PAGE 12).

Les options de configuration suivantes ont été choisies :

a) À l'étiquette principale (figure 4.10) :

a) Type de dispositif : « **USB Embedded Host** »

b) Mode de **Ping-Pong** : « Tous les points finaux(endpoint) »

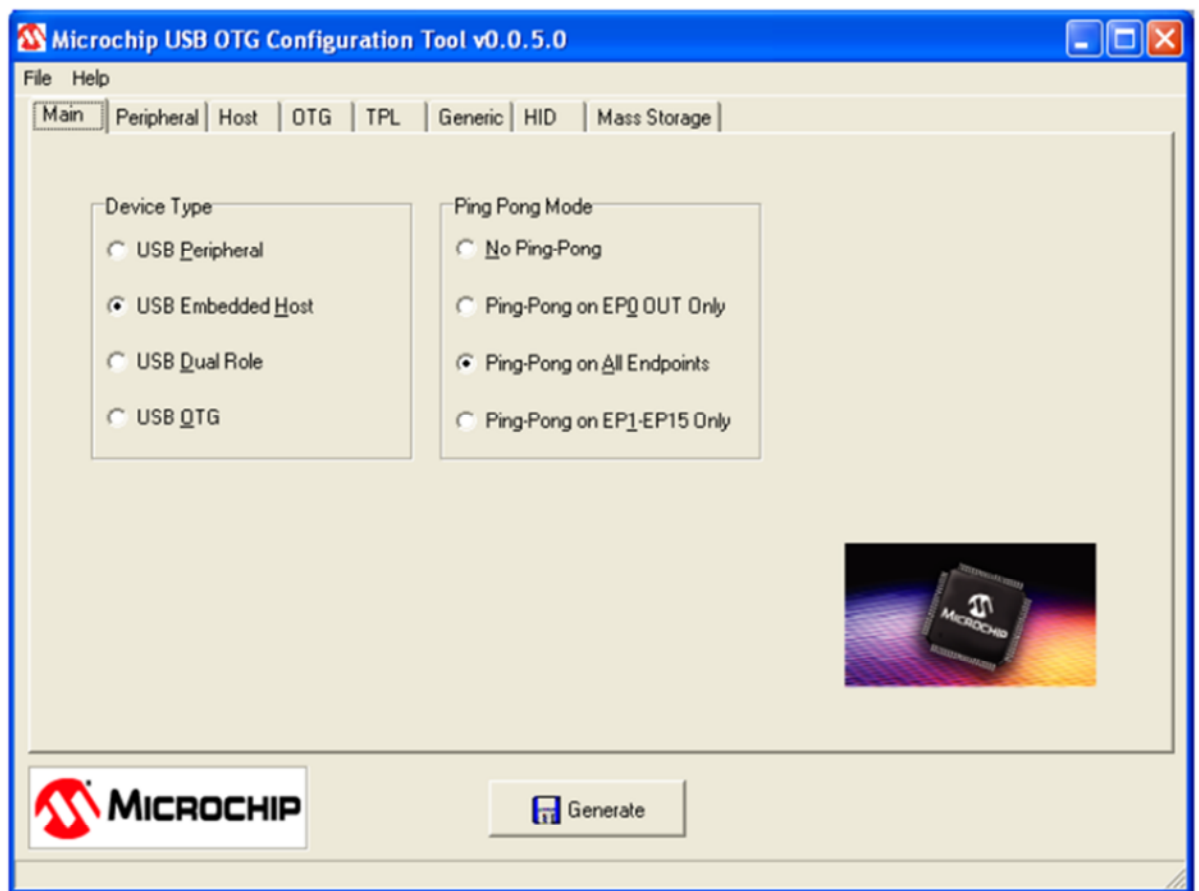


Figure 4.10

b) À l'étiquette d'hôte (figure 4.11) :

- a) Type de transfert : « **BulkTransfers** » seulement, avec **10.000 NAKs** autorisé (control Transfers sont également autorisé, avec textes de dialogue dans le gris)
- b) Attachez le temps de **Debounce : 250 ms**
- c) Nom de manipulateur d'événement d'application **USB_ApplicationEventHandler.**

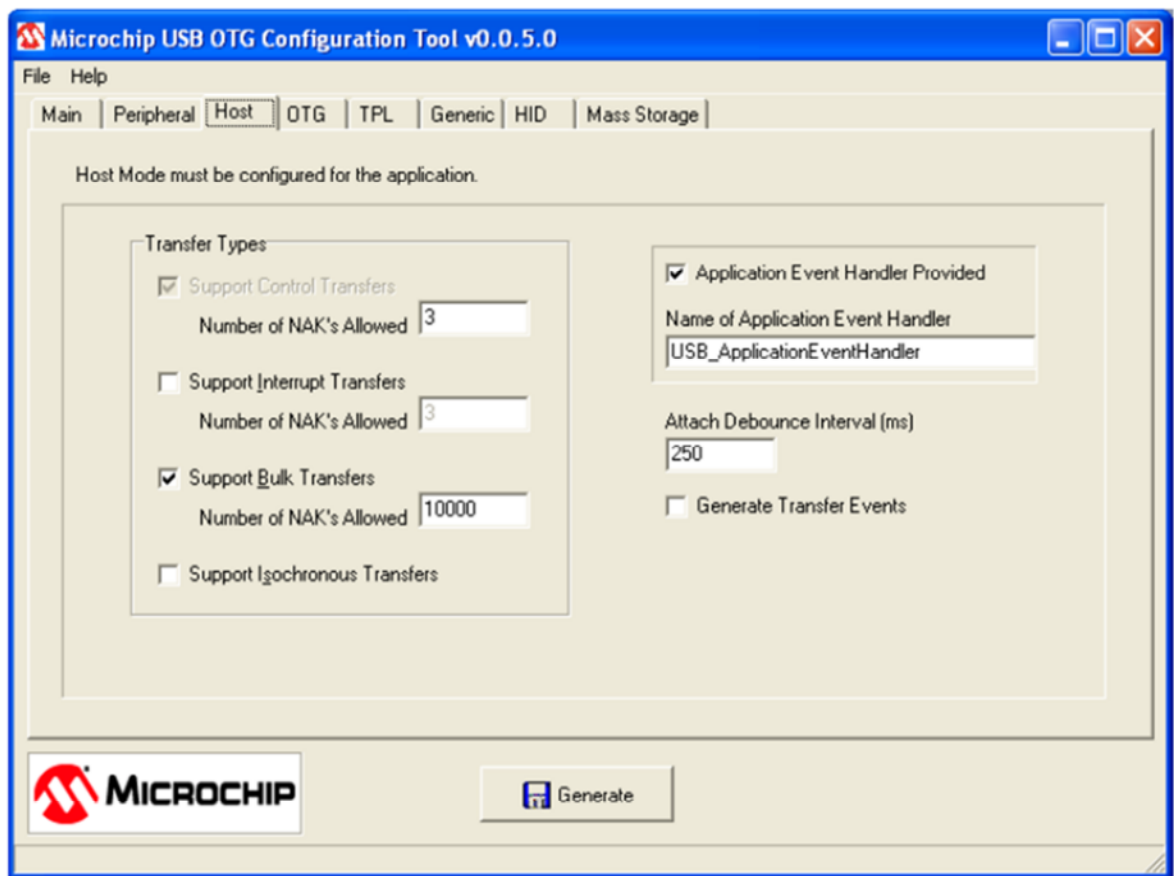


Figure 4.11

- c) À l'étiquette de TPL (liste périphérique visée) (figure 4.12):
- a) L'appui par l'intermédiaire de l'identification de classe ID est choisi
 - b) driver de client: Mémoire de masse
 - c) Classe: Mémoire de masse (0x08)
 - d) Sous-classe: Commande de SCSI réglée (0x06)
 - e) Protocole: transport réservé au Bulk Transfers (0x50)
 - f) Configuration initiale: 0
 - g) flag d'initialisation: 0

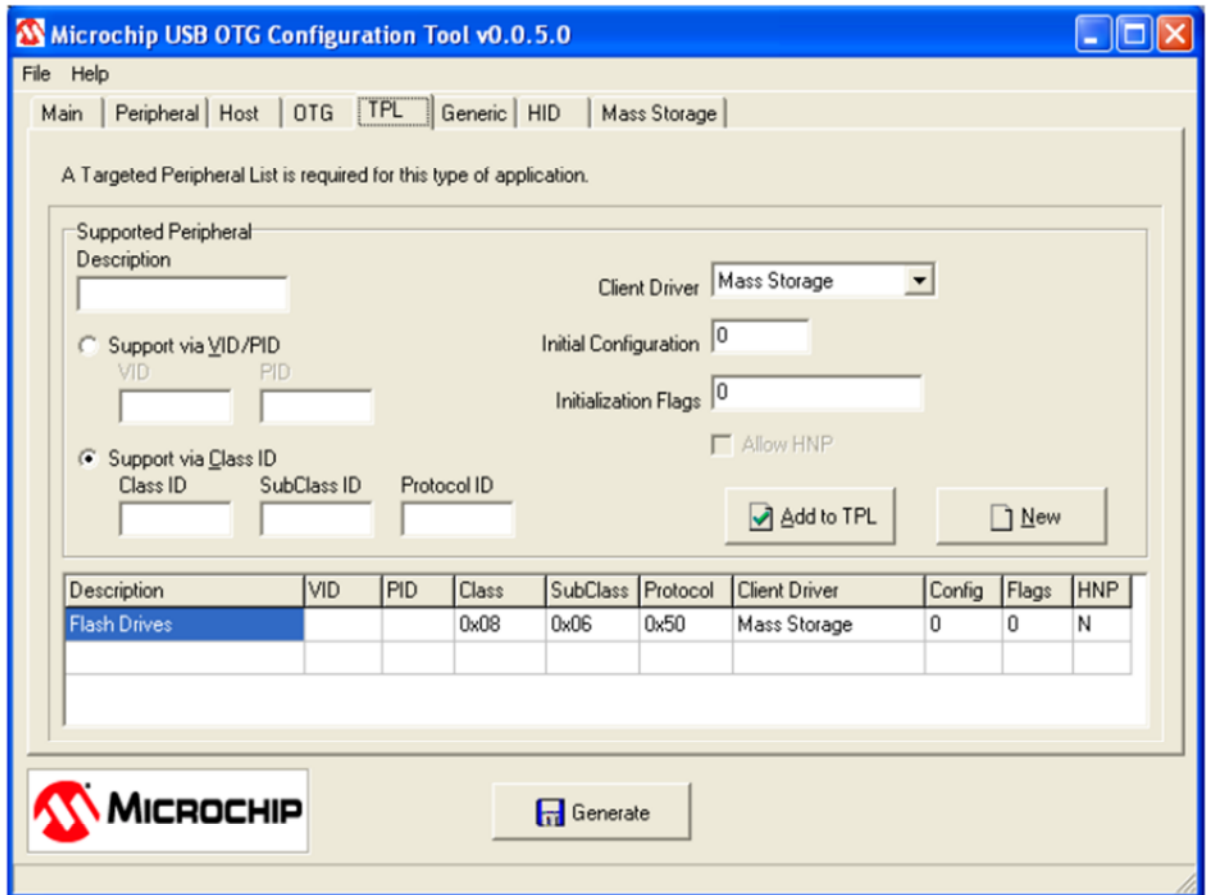


Figure 4.12

d) À l'étiquette de mémoire de masse (figure 4.13):

- a) 'Mass Storage Client is used in Host mode' est choisi.
- b) Dans « les médias de stockage », l'option « **interface SCSI** » est choisie

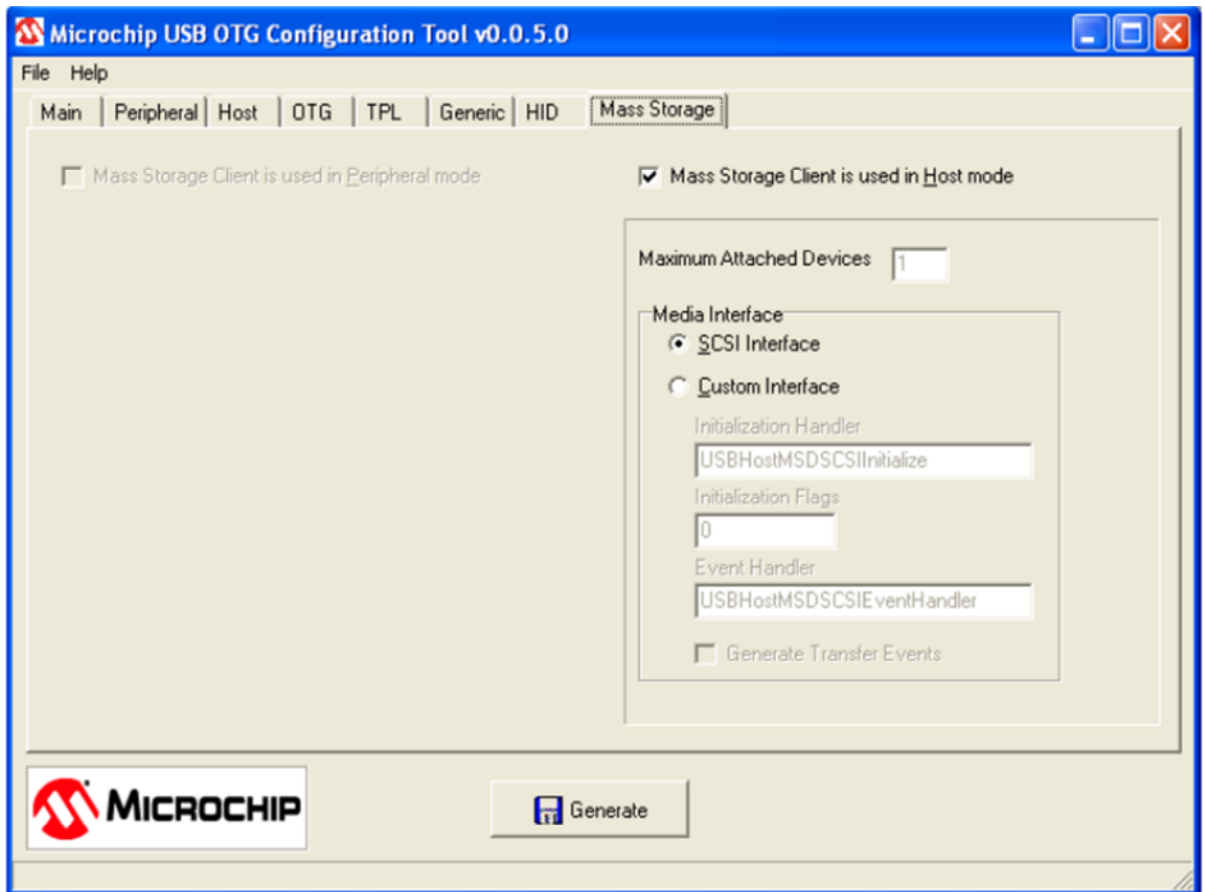


Figure4.13

4.3.3 Organigramme du programme global

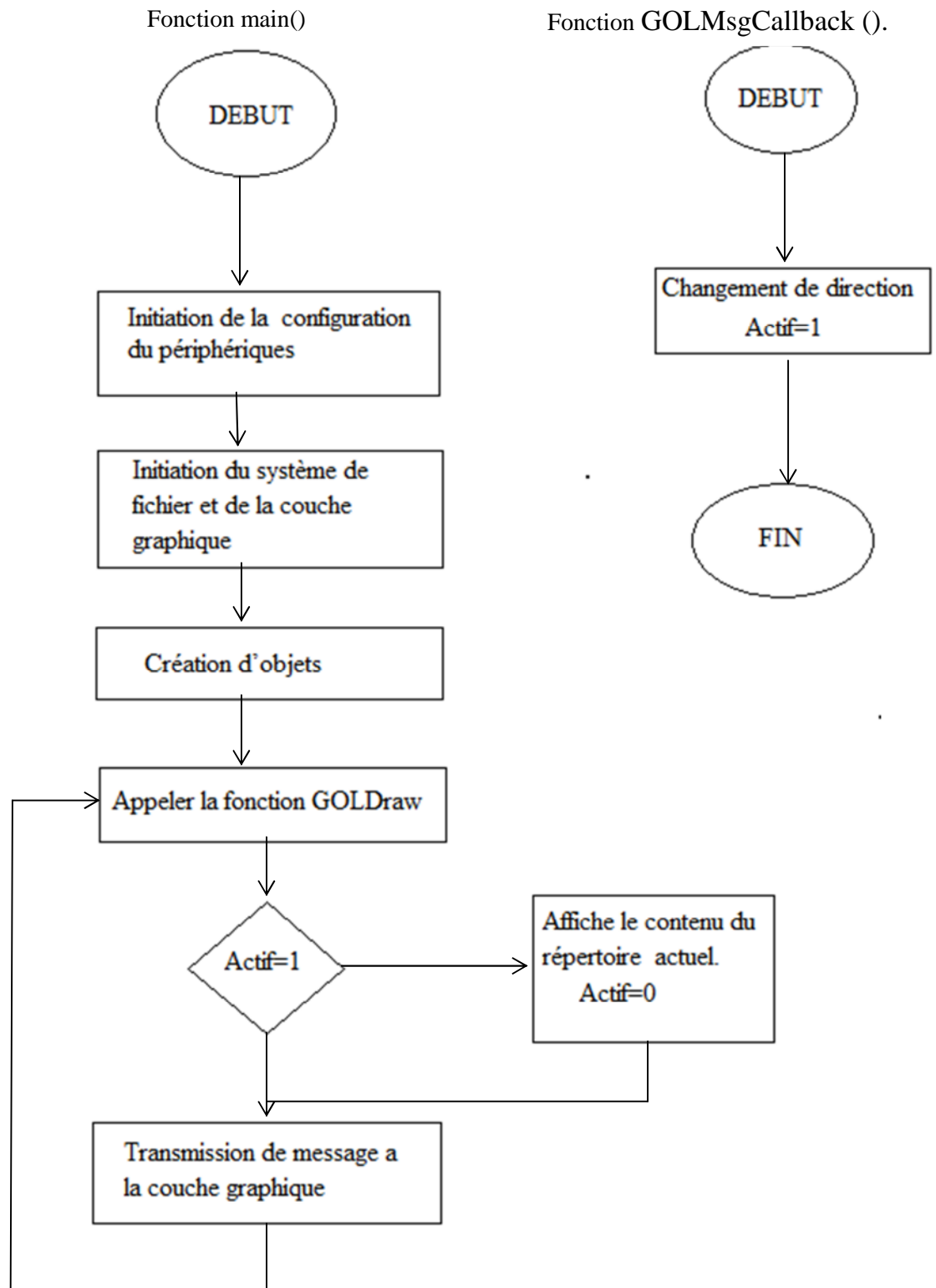


Figure 4.14

4.4 conclusion

Malgré les petits changement effectués au cour des différentes étapes, le projet a abouti grâce à la réalisation :

- La carte électronique
- Le développement du programme sur PIC32MX795F512H