

## Objectifs

- Comprendre le fonctionnement de l'écran LCD
- Initialiser l'écran
- Effacer l'écran

## Déclaration des entrées/sorties

Ajouter dans le fichier **GameGir1.h** les entrées / sorties qui permettent de commander l'afficheur LCD en respectant les noms suivants :

Sélection du contrôleur A	<b>LCD_CSA</b>
Sélection du contrôleur B	<b>LCD_CSB</b>
Sélection du registre	<b>LCD_RS</b>
Lecture / écriture	<b>LCD_RW</b>
Enable	<b>LCD_Enable</b>
Port de données	<b>LCD_DATA</b>
Reset	<b>LCD_Reset</b>
Rétro éclairage	<b>LCD_BackLight</b>

Dans la fonction **void ConfigProcessor(void)** ajouter les broches de l'afficheur LCD en les configurant toutes en sorties.

## Rétro éclairage

Ecrire un programme principal qui fait clignoter une led en même temps que la broche qui commande le rétro éclairage. Programmer et tester. Ajuster le potentiomètre si nécessaire.

## Initialisation

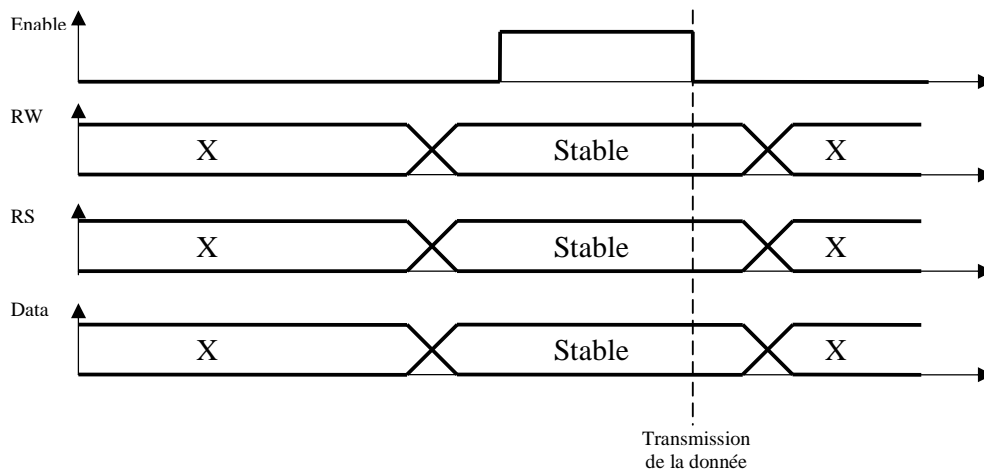
Créer une bibliothèque LCD composée d'un fichier **LCD.H** et d'un fichier **LCD.c**. Définir la macro **\_\_LCD\_\_** dans le premier fichier afin d'éviter les appels multiples à la bibliothèque.

Créer une fonction **void LCD\_Init(void)** qui sélectionne les deux contrôleurs (CSA=CSB=1), puis désactive la broche Enable (Enable=0) avant de mettre le reset à 0 pendant 100 cycles pour enfin le remettre à 1.

Vérifier que vous n'avez pas d'erreur de compilation, mais ne transférer pas le programme.

### Instructions

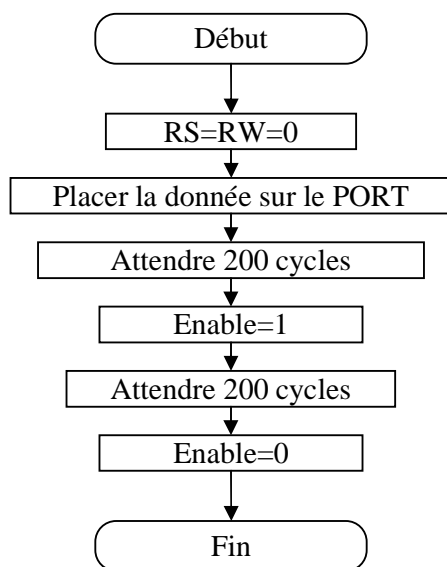
Les chronogrammes ci-dessous montrent un cycle du dialogue entre le microcontrôleur et l'écran :



L'afficheur LCD est commandé selon deux modes.

- Le premier permet d'écrire dans les registres de configuration, il s'agit du mode instruction (RS=0). Ce mode peut être utilisé en écriture (RW=0) ou en lecture (RW=1) pour connaître l'état de l'afficheur.
- Le second permet d'écrire (RW=0) et de lire (RW=1) les caractères sur l'écran, il s'agit du mode donnée (RS=1).

Ecrire la fonction **void LCD\_Write\_Command (unsigned char Data)** qui va permettre d'envoyer l'instruction **Data** vers l'afficheur LCD selon l'organigramme suivant :



A la fin de la fonction d'initialisation, ajouter l'envoi de l'instruction qui permet d'allumer l'afficheur grâce à la fonction que vous venez d'écrire. Utiliser l'extrait de la documentation fourni en annexe pour déterminer la valeur à envoyer.

Initialiser l'écran dans le programme principal et tester. Vous devez voir l'écran s'allumer.

### Données

Ecrire la fonction **void LCD\_Send\_Data (unsigned char Data)** qui va permettre d'envoyer la donnée **Data** vers l'afficheur LCD. Le code de cette fonction est strictement identique à la précédente à un caractère près.

Ajouter dans la boucle du programme principal l'envoi d'une donnée quelconque et vérifier que l'affichage fonctionne.

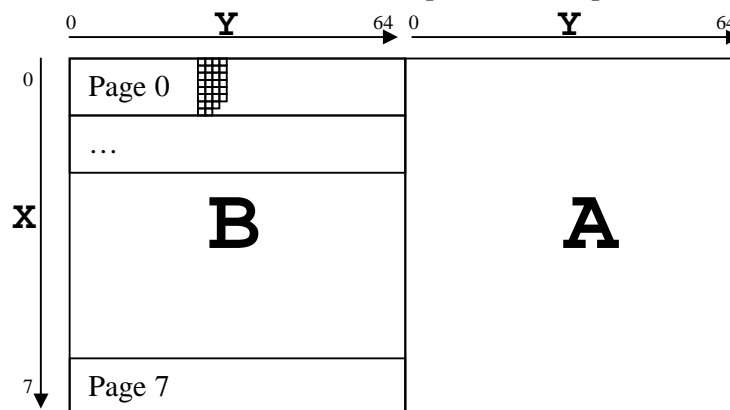
### Fonctionnement de l'écran

L'écran est présenté comme un écran graphique de 128 x 64 pixels, mais en réalité pour des raisons technologiques l'écran est composé de 2 contrôleurs (A et B) qui sont sélectionnées par CSA et CSB. Taper le programme principal suivant et tester toutes les combinaisons possibles de CSA et CSB :

```
int i=0;
ConfigProcessor(); // Configuration du micro
LCD_Init();        // Initialisation de l'écran LCD
LCD_BackLight=1;   // Allume le rétro éclairage
LCD_CSA=1;         // Sélectionne les contrôleurs
LCD_CSB=0;
while (1) LCD_Send_Data (i++); // Affiche une donnée
```

L'écran est donc divisé en deux parties de 64 x 64 pixels.

Chaque partie est composée de 8 pages de 64 x 8 pixels ou 64 x 1 octet. L'écriture d'une donnée sur l'écran permet de commander directement chaque octet. Lorsqu'un octet est écrit, le contrôleur passe directement au suivant. Notez que cela n'est pas valable pour les pages.

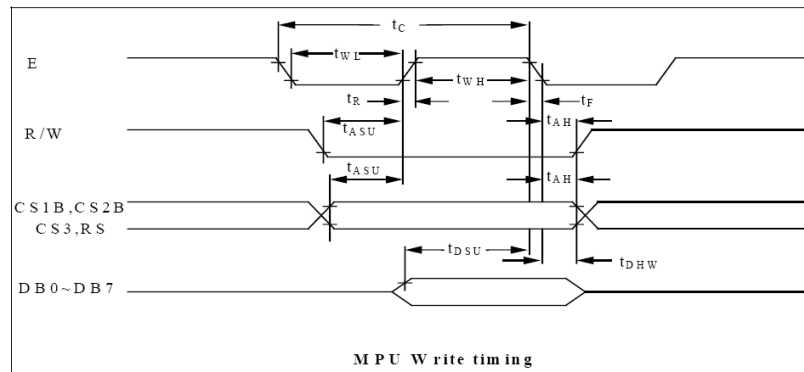


### Effacer l'écran

Ecrire une fonction **void LCD\_Clear (void)** qui permet d'effacer l'écran. Pour cela, sélectionner les deux contrôleurs et balayer les 8 pages en envoyant à chaque fois 64 octets nuls. Lorsque la fonction est opérationnelle, insérer un appel à cette fonction dans la routine d'initialisation juste avant l'allumage de l'écran.

### Extraits de la documentation

RS	R/W	Function
L	L	Instruction
	H	Status read (busy check)
H	L	Data write (from input register to display data RAM )
	H	Data read (from display data RAM to output register)



Instruction	RS	RW	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function
Read Display Date	1	1	Read data								Reads data (DB[7:0]) from display data RAM to the data bus.
Write Display Date	1	0	Write data								Writes data (DB[7:0]) into the DDRAM. After writing instruction, Y address is incremented by 1 automatically
Status Read	0	1	Busy	0	ON/OFF	Re-set	0	0	0	0	Reads the internal status BUSY 0: Ready 1: In operation ON/OFF 0: Display ON 1: Display OFF RESET 0: Normal 1: Reset
Set Address (Y address)	0	0	0	1	Y address (0~63)						Sets the Y address at the column address counter
Set Display Start Line	0	0	1	1	Display start line (0~63)						Indicates the Display Data RAM displayed at the top of the screen.
Set Address (X address)	0	0	1	0	1	1	1	Page (0~7)			Sets the X address at the X address register.
Display On/off	0	0	0	0	1	1	1	1	1	0/1	Controls the display ON or OFF. The internal status and the DDRAM data is not affected. 0: OFF, 1: ON