





# Modules Afficheurs LCD Alphanumériques

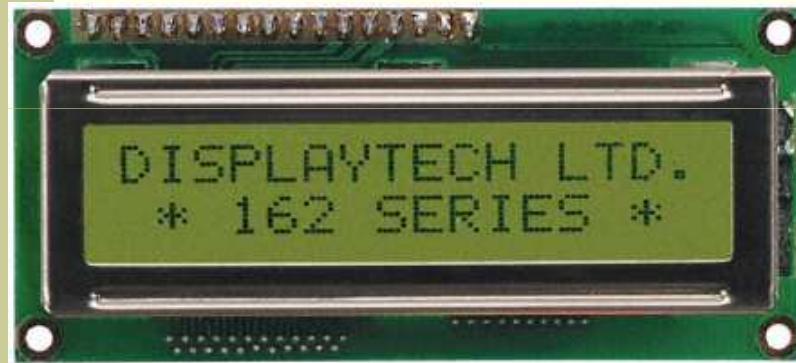
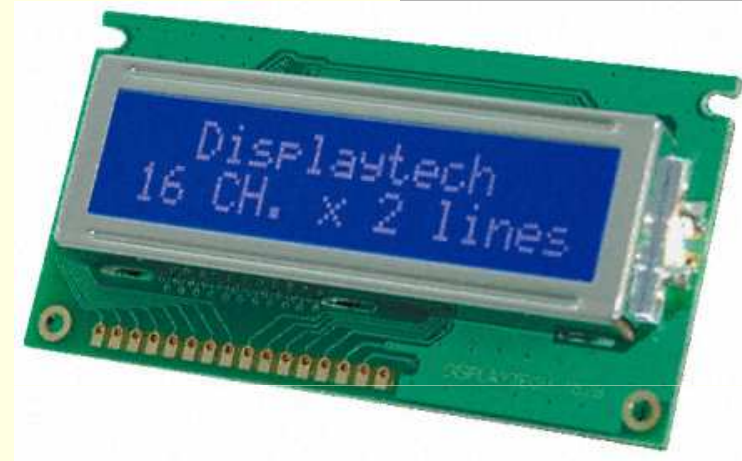


Matériels et Logiciels (8 bits)



# Différents Types de Modules Afficheurs LCD

Afficheur, LCD, alphanumérique, 16 x 2,  
STN, négatif bleu, 84 x 4.4mm,



Afficheur, LCD, alphanumérique,  
20 x 4, FSTN, négatif, 98 x  
60mm

Afficheur, LCD, alphanumérique, 16 x 2,  
STN, sans led, 80 x 36,



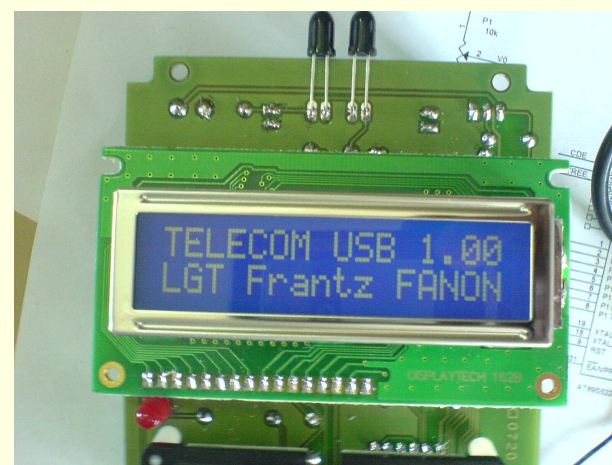
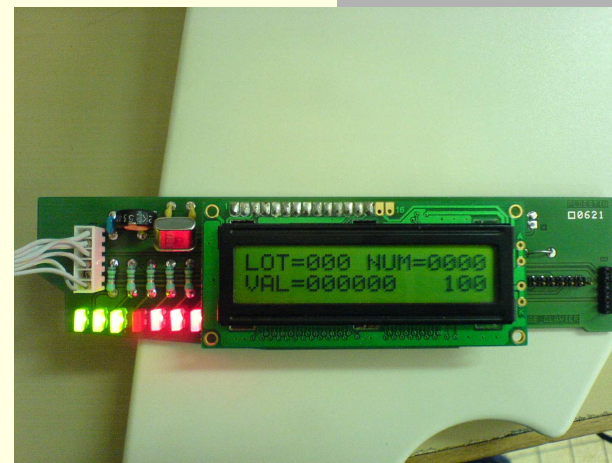
# Différents Modes d'Eclairage

---

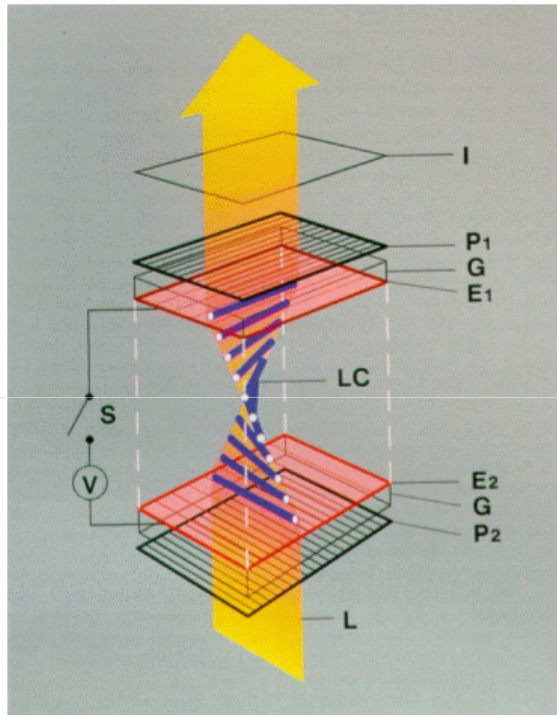
- Réflectifs = Ecran qui utilise la lumière ambiante pour fournir l'image dans des lieux très éclairés ou en extérieur.
- Transmissif = Ecran équipé d'un dispositif de rétro-éclairage pour une utilisation dans un endroit peu éclairé.
- Transflectifs = Ecran équipé d'un dispositif de rétro-éclairage (transmissif) et qui utilise, également, la lumière ambiante (réflectif).
- Positif = Caractères sombres sur fond clair.
- Négatif = Caractères clairs sur fond sombre.

# Deux Exemples d'Utilisation

- Nouveau panneau de contrôle Compteur de Billets,
- Télécommande Universelle.



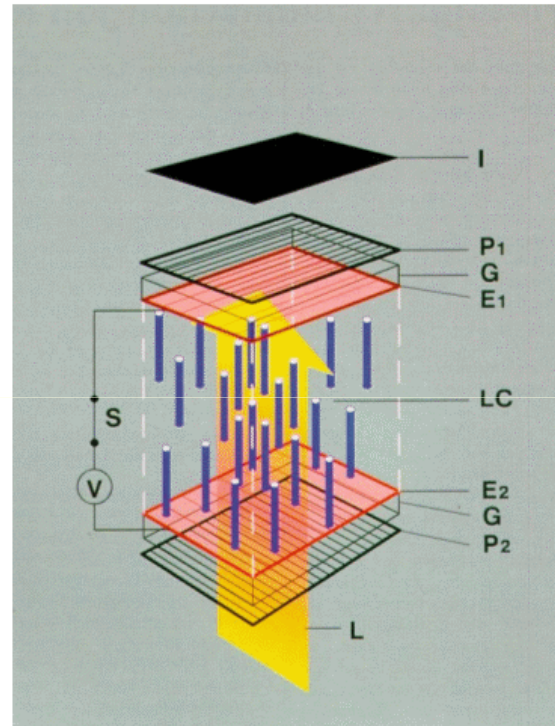
# Fonctionnement d'un segment LCD



La lumière (L) est polarisée par le filtre (P2).

Elle suit la rotation de phase des cristaux liquides (LC).

Grâce à cette rotation, la lumière traverse aussi le filtre croisé (P1).



La rotation de phase des cristaux liquides (LC) est momentanément désorganisée par un champ électrique entre les électrodes E1 et E2.

La lumière (L) est polarisée par le filtre (P2).

La lumière ne peut pas traverser le filtre croisé (P1).

- P1/P2 : filtres polarisés croisés.
- E1/E2 : électrodes de commande.
- L : Faisceau de lumière émis par l'éclairage arrière.
- LC : Cristaux liquides.
- I : Observation...

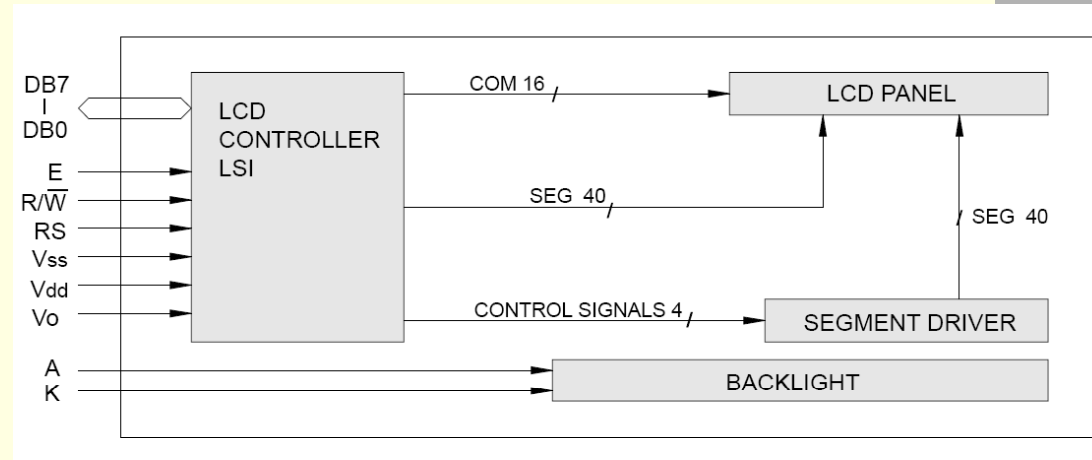
# Quelques Définitions

---

- Nématique = Etat moléculaire d'un cristal liquide en l'absence d'influence extérieure.
- TN (Twisted Nematic) = nématique en hélice.
- STN (Super Twisted Nematic) = nématique en hélice à multiplexage élevé.
- DSTN (Double Super Twisted Nematic) = Ecran LCD utilisant 2 panneaux STN pour un meilleur contraste.
- FSTN = (Film Super Twisted Nematic) Ecran avec un filtre en polymère pour un meilleur contraste.

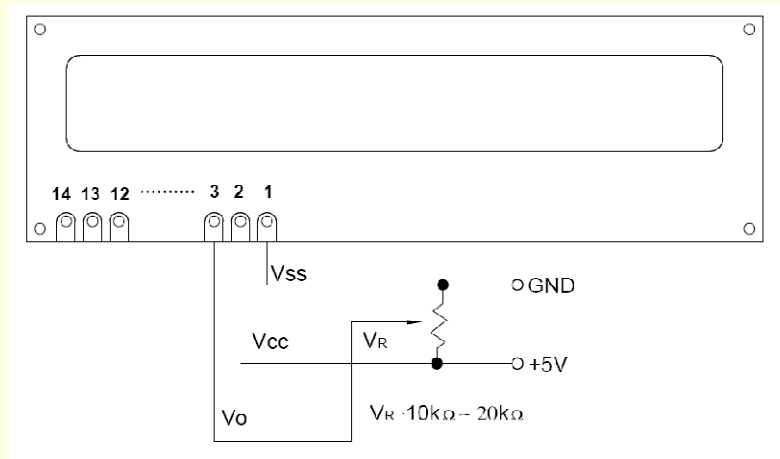


# Structure Interne d'un Module



- LCD Panel = Matrice de pixels LCD ;
- LCD controller LSI = Contrôleur intelligent LSI ;
- Segment Driver = Commande de segments étendue ;
- Backlight = Eclairage arrière optionnel (LED ou CCFL).

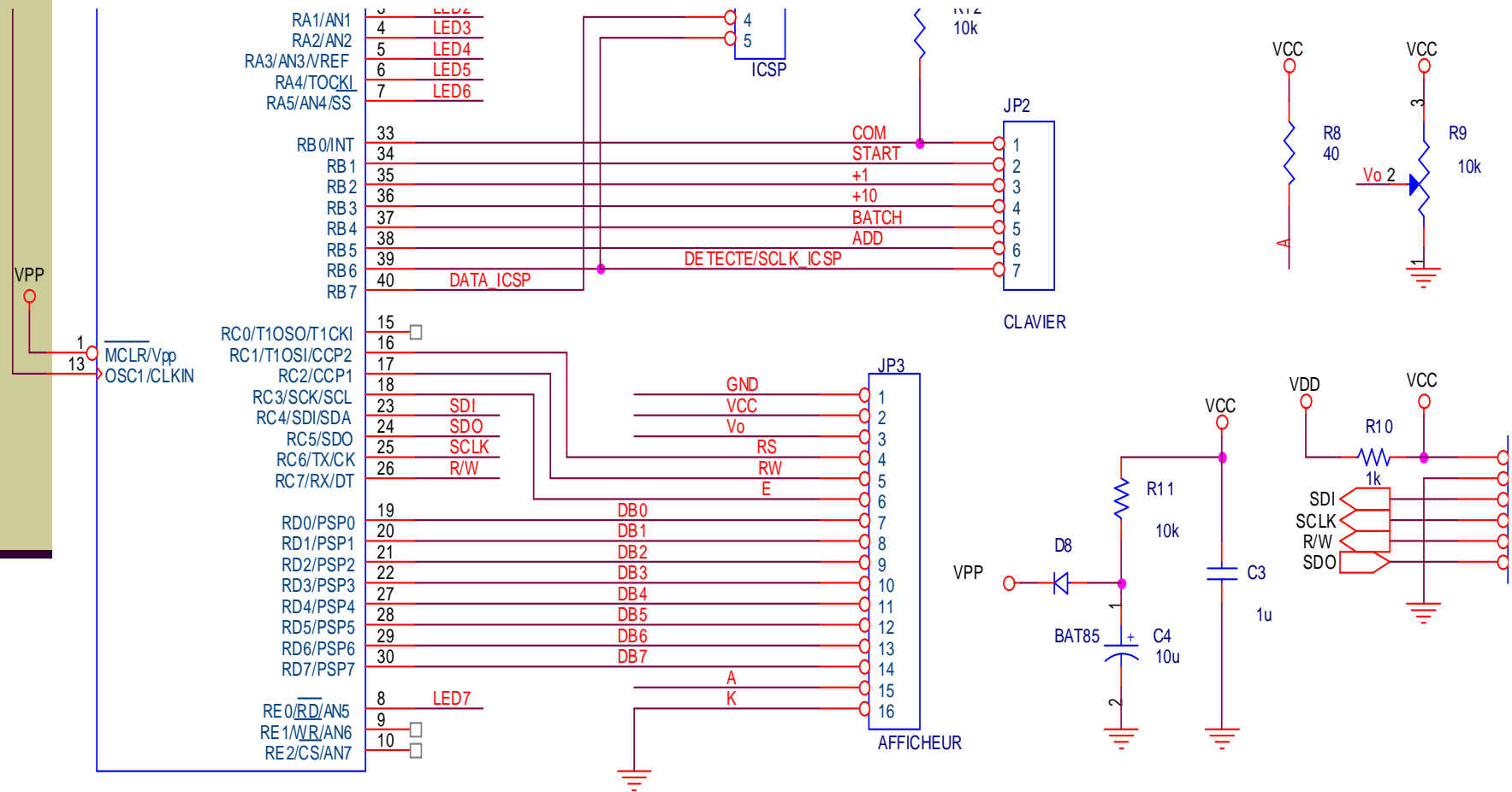
# Connecteur



| N° Broche | NOM              | Description  |
|-----------|------------------|--|
| 1         | <b>VSS</b>       | Masse  |
| 2         | <b>VDD</b>       | Alimentation positive (V+)                         |
| 3         | <b>VO</b>        | Contraste réglable par potentiomètre *             |
| 4         | <b>RS</b>        | Sélection de registre (0= instruction; 1 = donnée) |
| 5         | <b>R/W ou RD</b> | Lecture ou écriture (1= lecture; 0=écriture)       |
| 6         | <b>E</b>         | Enable (validation, actif au niveau haut)          |
| 7         | <b>D0</b>        | Bit 0 du bus de données                            |
| 8         | <b>D1</b>        | Bit 1 du bus de données                            |
| 9         | <b>D2</b>        | Bit 2 du bus de données                            |
| 10        | <b>D3</b>        | Bit 3 du bus de données                            |
| 11        | <b>D4</b>        | Bit 4 du bus de données                            |
| 12        | <b>D5</b>        | Bit 5 du bus de données                            |
| 13        | <b>D6</b>        | Bit 6 du bus de données                            |
| 14        | <b>D7</b>        | Bit 7 du bus de données                            |

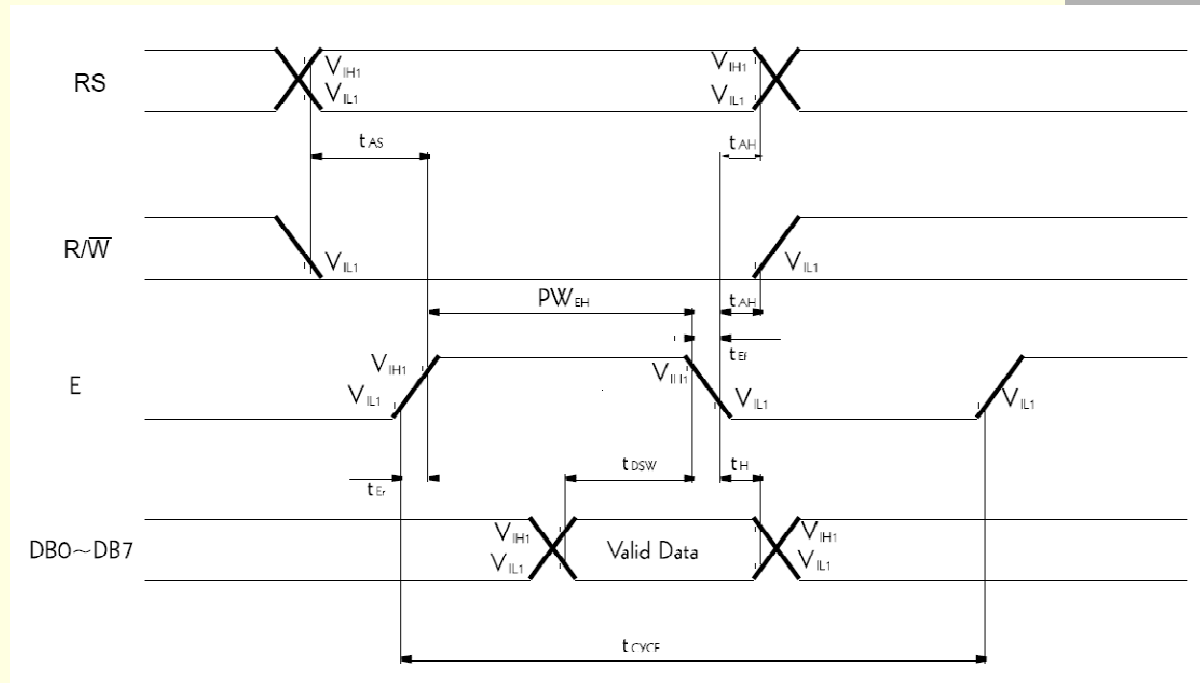


# Raccordement (8 bits)



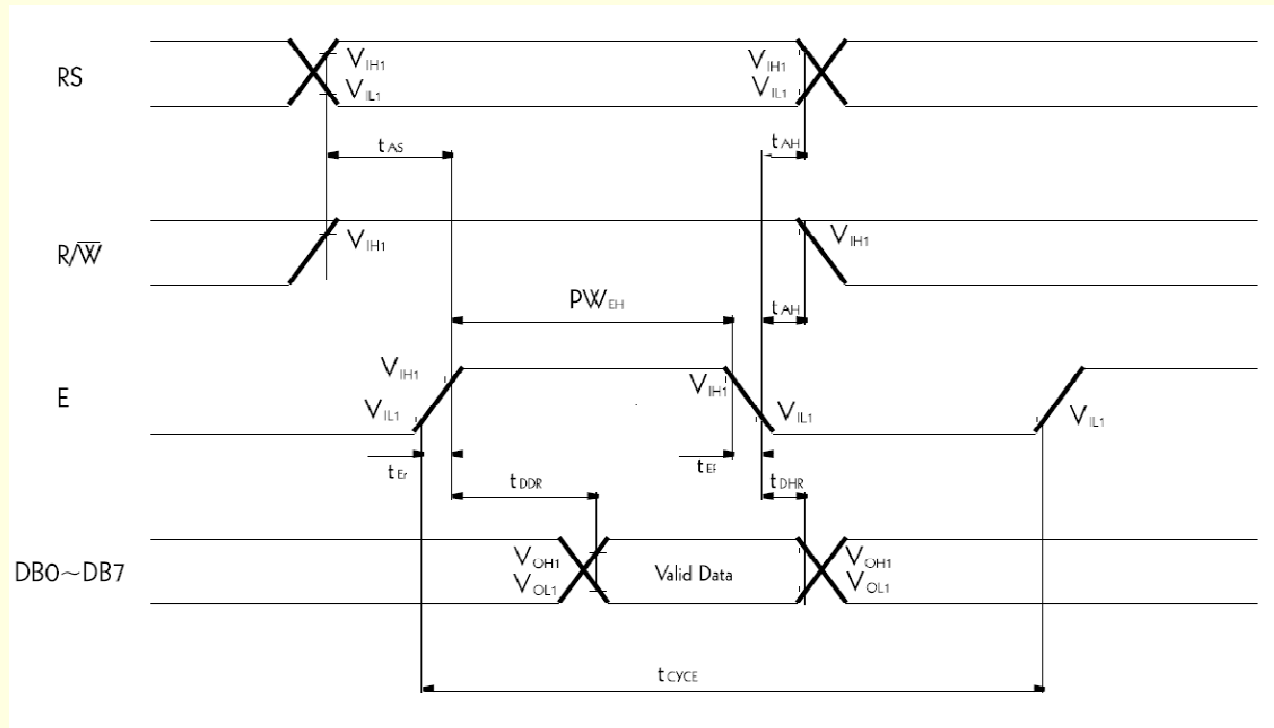
PIC16F877A

# Cycle d'écriture (8 bits)



- RS sélectionne le type d'écriture : Cde ou Data.
- R/W passe à « 0 » pour signifier l'écriture.
- E confirme la mémorisation des données à chaque impulsion (front descendant).

# Cycle de Lecture (8 bits)

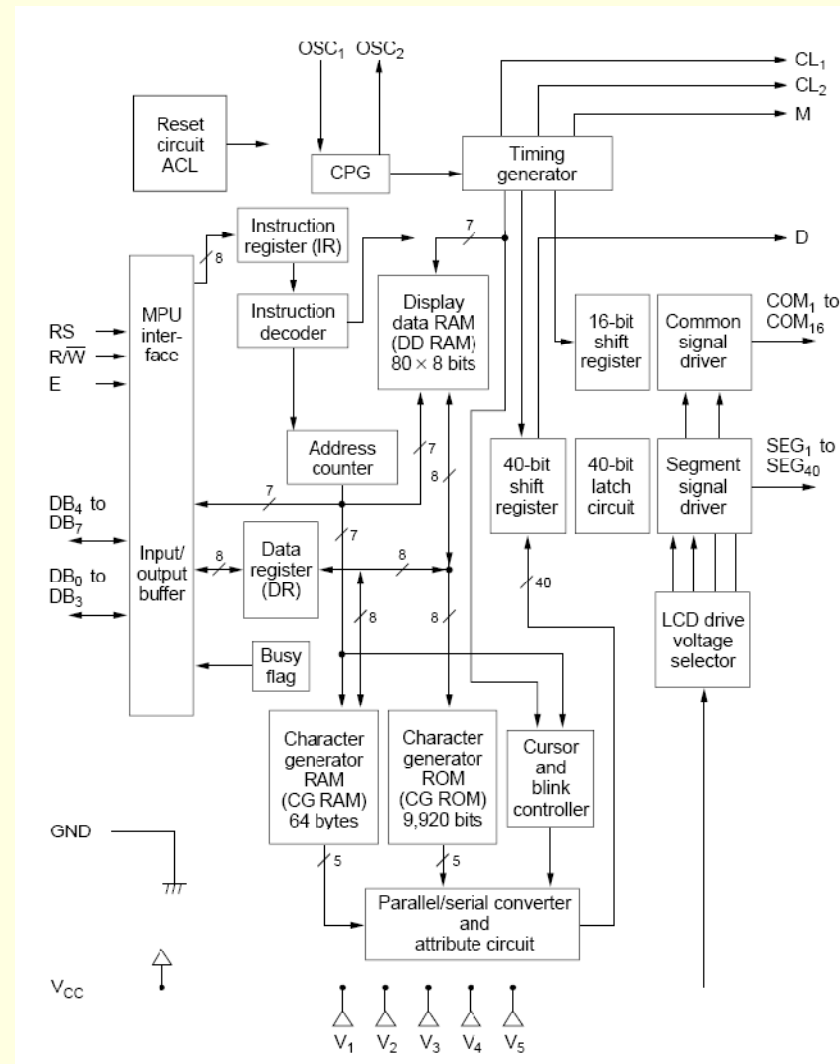


- RS sélectionne le type de lecture : Statut ou Data.
- R/W passe à « 1 » pour signifier la lecture.
- E confirme la sortie des données pendant l'état « 1 ».

# Configuration des Transferts

| RS | R/W | Fonctions  |
|----|-----|--|
| 0  | 0   | Ecriture d'une commande dans le registre d'instruction IR (clear, etc.)          |
| 0  | 1   | Lecture de Statut<br>BF (busy flag = DB7)<br>AC (compteur d'adresse = DB0 à DB6) |
| 1  | 0   | Ecriture d'une donnée dans la DDRAM ou CGRAM                                     |
| 1  | 1   | Lecture d'une donnée de la DDRAM or CGRAM  |

# Structure Interne Contrôleur LCD



- Tous les échanges transitent par le buffer d'entrée/sortie.
- Les instructions transitent par l'intermédiaire du registre d'instructions (IR).
- Les données transitent par le registre de données (DR).
- La DDRAM mémorise les données d'affichage en code ASCII. Ici 2 lignes de 16 caractères.
- La CGROM génère un motif, de 5x8 ou 5x10 points, pour chaque code ASCII.
- L'utilisateur peut définir des caractères personnalisés dans la CGRAM. On utilise une matrice de 5x8 ou 5x10 points.

# Display Data RAM (DDRAM)

---

- La DDRAM mémorise les données d'affichage en code ASCII. Ici 2 lignes de 16 caractères.

|                   |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |
|-------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Character located | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| DDRAMaddress      | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
| DDRAMaddress      | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F |





# Character Generator RAM (CGRAM)

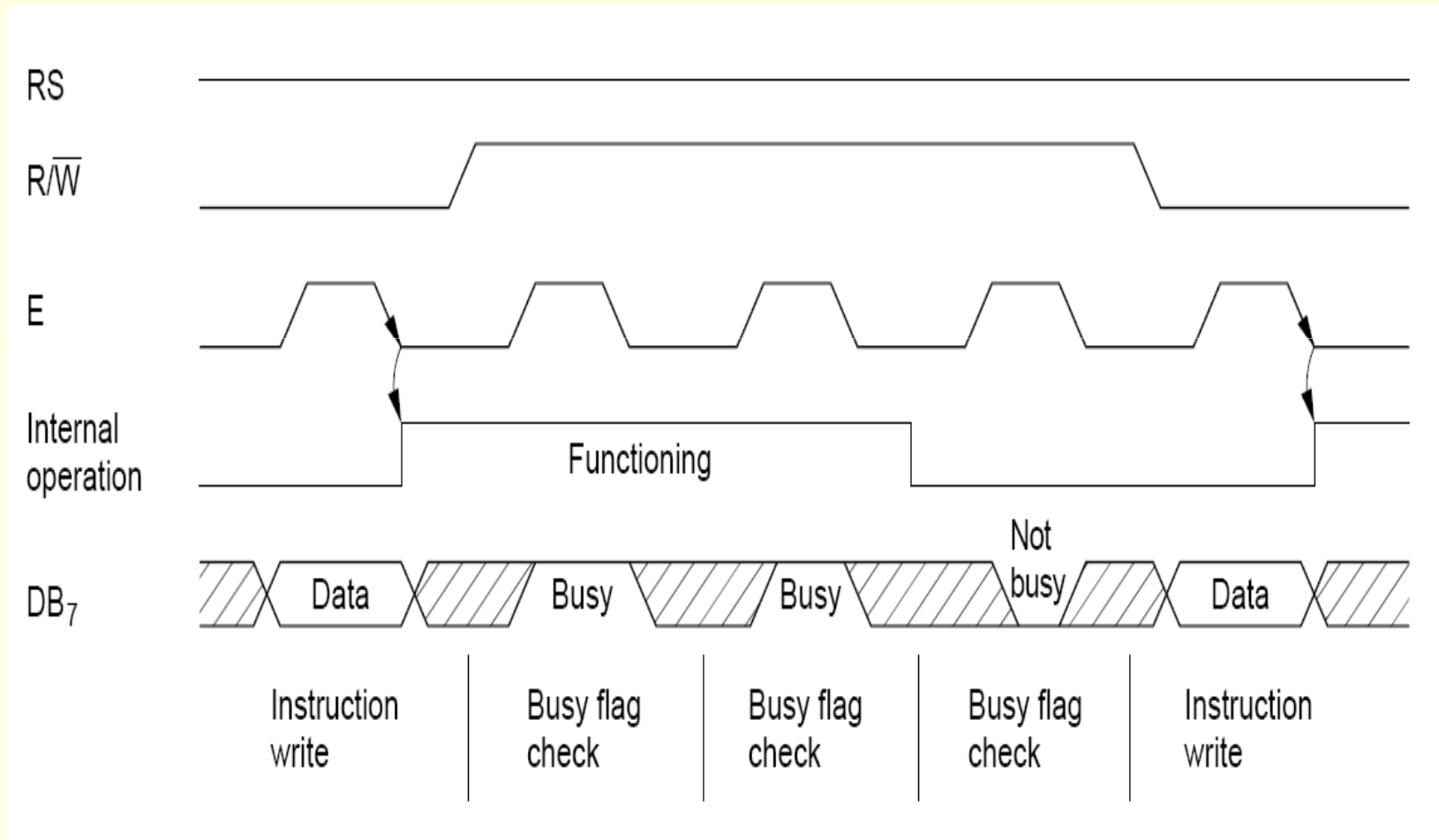
- L'utilisateur peut définir des caractères personnalisés dans la CGRAM. On utilise une matrice de 5x8 ou 5x10 points.

| Character Codes<br>( DDRAM data ) |   |   |   |     |   |   |   | CGRAM Address |  |     |   |   | Character Patterns<br>( CGRAM data ) |   |   |           |     |  |   |     |   |   |   |                           |   |   |   |   |
|-----------------------------------|---|---|---|-----|---|---|---|---------------|--|-----|---|---|--------------------------------------|---|---|-----------|-----|--|---|-----|---|---|---|---------------------------|---|---|---|---|
| 7                                 | 6 | 5 | 4 | 3   | 2 | 1 | 0 | 5             |  | 4   |   |   | 3                                    |   | 2 |           | 1   |  | 0 |     | 7 | 6 | 5 | 4                         | 3 | 2 | 1 | 0 |
| High                              |   |   |   | Low |   |   |   | High          |  | Low |   |   | High                                 |   |   |           | Low |  |   |     |   |   |   |                           |   |   |   |   |
| 0 0 0 0 * 0 0 0                   |   |   |   |     |   |   |   | 0 0 0         |  | 0   | 0 | 0 | *                                    | * | * | 0 0 0     |     |  |   | 0   |   |   |   | Character<br>pattern( 1 ) |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 0 | 1 | *                                    | * | * | 0 0 0     |     |  |   |     |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 1 | 0 | *                                    | * | * | 0 0 0     |     |  |   |     |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 1 | 1 | *                                    | * | * | 0         |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 0 | 0 | *                                    | * | * | 0         |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 0 | 1 | *                                    | * | * | 0 0       |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 1 | 0 | *                                    | * | * | 0 0 0     |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 1 | 1 | *                                    | * | * | 0 0 0 0 0 |     |  |   | 0   |   |   |   |                           |   |   |   |   |
| 0 0 0 0 * 0 0 1                   |   |   |   |     |   |   |   | 0 0 1         |  | 0   | 0 | 0 | *                                    | * | * | 0 0 0 0   |     |  |   | 0   |   |   |   | Cursor<br>pattern         |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 0 | 1 | *                                    | * | * | 0         |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 1 | 0 | *                                    | * | * | 0         |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 1 | 1 | *                                    | * | * | 0 0       |     |  |   | 0 0 |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 0 | 0 | *                                    | * | * | 0 0       |     |  |   | 0 0 |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 0 | 1 | *                                    | * | * | 0 0       |     |  |   | 0 0 |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 1 | 0 | *                                    | * | * | 0 0       |     |  |   | 0 0 |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 1   | 1 | 1 | *                                    | * | * | 0 0 0 0 0 |     |  |   | 0   |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 0 | 0 | *                                    | * | * |           |     |  |   |     |   |   |   |                           |   |   |   |   |
|                                   |   |   |   |     |   |   |   |               |  | 0   | 0 | 1 | *                                    | * | * |           |     |  |   |     |   |   |   |                           |   |   |   |   |

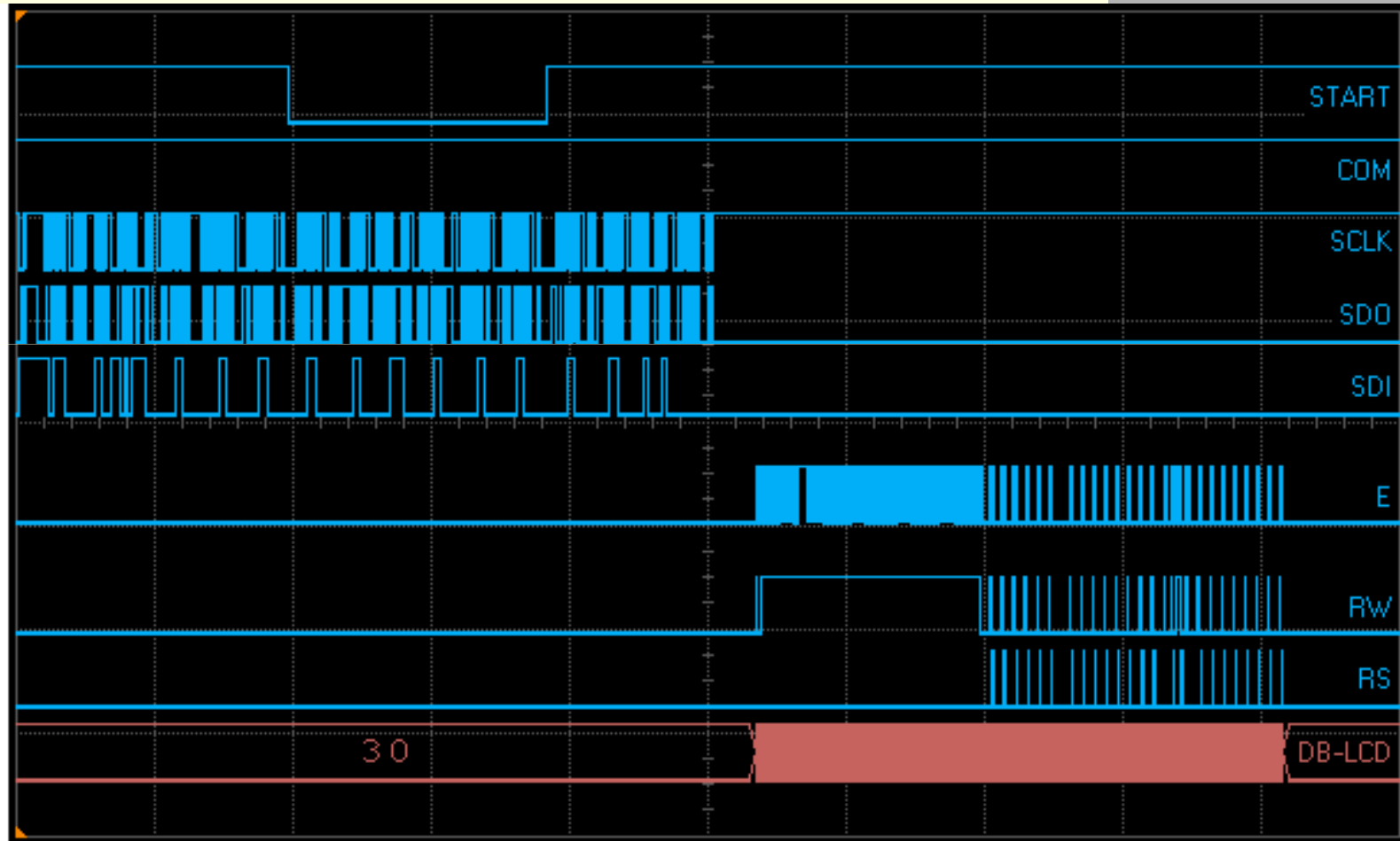
# Table de Commande

| Commande                                     | RS | R/W | D7      | D6  | D5  | D4 | D3 | D2 | D1   | D0   | Description   |
|--|----|-----|---------|-----|-----|----|----|----|--|--|---|
| Effacement                                   | 0  | 0   | 0       | 0   | 0   | 0  | 0  | 0  | 0  | 1  | Efface l'écran et positionne le curseur à 0 (haut,gauche)   |
| Retour Curseur                               | 0  | 0   | 0       | 0   | 0   | 0  | 0  | 0  | 1  | X  | Positionne le curseur à 0 (haut,gauche)   |
| Mode de décalage                             | 0  | 0   | 0       | 0   | 0   | 0  | 0  | 1  | ID   | S  | Définit la direction de déplacement du curseur et si le texte doit suivre le curseur<br>ID : 1 = incremente adresse DD-RAM ; 0 = decremente<br>S : 1 = Décale l' affichage; 0 = ne décale pas |
| Afficheur On/Off,<br>curseur et clignotement | 0  | 0   | 0       | 0   | 0   | 0  | 1  | D  | C  | B  | D : 1 = Affichage On ; 0 = Affichage Off<br>C : 1 = Curseur en service ; 0 = Curseur Hors service<br>B : 1 = Curseur clignotant ; 0 = Curseur Fixe  |
| Décalage Affichage et<br>curseur             | 0  | 0   | 0       | 0   | 0   | 1  | SC | RL | X  | X  | SC : 1 = Décale l' afficheur ; 0 = Décale le curseur<br>RL : 1 = Décalage à droite; 0 = Décalage à gauche   |
| Initialisation                               | 0  | 0   | 0       | 0   | 1   | DL | N  | X  | X  | X  | DL : 1 = Adressage 8 bits ; 0 = Adressage 4 Bits<br>N : 1 = Afficheur 2 lignes ; 0 = Afficheur 1 ligne  |
| Adresse CG-RAM                               | 0  | 0   | 0       | 1   | ACG |    |    |    |  | Définit l' adresse de la CG-RAM dans le compteur d' adresse. |   |
| Adresse DD-RAM                               | 0  | 0   | 1       | ADD |     |    |    |    | Définit l' adresse de la DD-RAM dans le compteur d' adresse.                     |  |   |
| Busy-flag et adresse                         | 0  | 1   | BF      | AC  |     |    |    |    | BF : Lit le drapeau d' occupation<br>AC : Lit le contenu du compteur d' adresse. |  |   |
| Écrire dans DD-RAM ou<br>CG-RAM              | 1  | 0   | Données |     |     |    |    |    |  |  |   |
| Lire dans DD-RAM ou<br>CG-RAM                | 1  | 1   | Données |     |     |    |    |    |  |  |   |

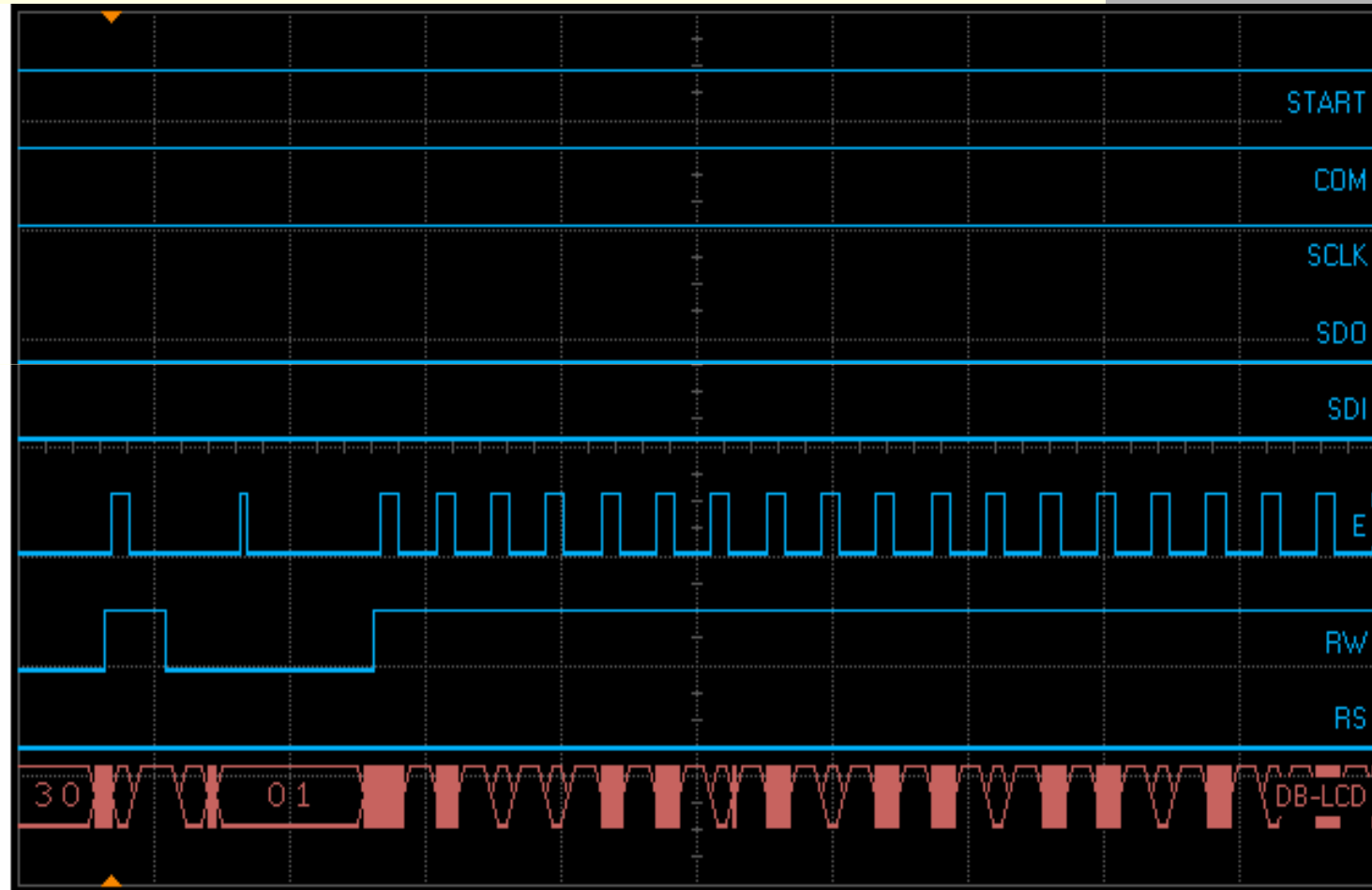
# Attente de BF (Busy Flag)



# Chronogrammes $\mu$ C/Module LCD



# Détail Echange $\mu$ C/Module LCD

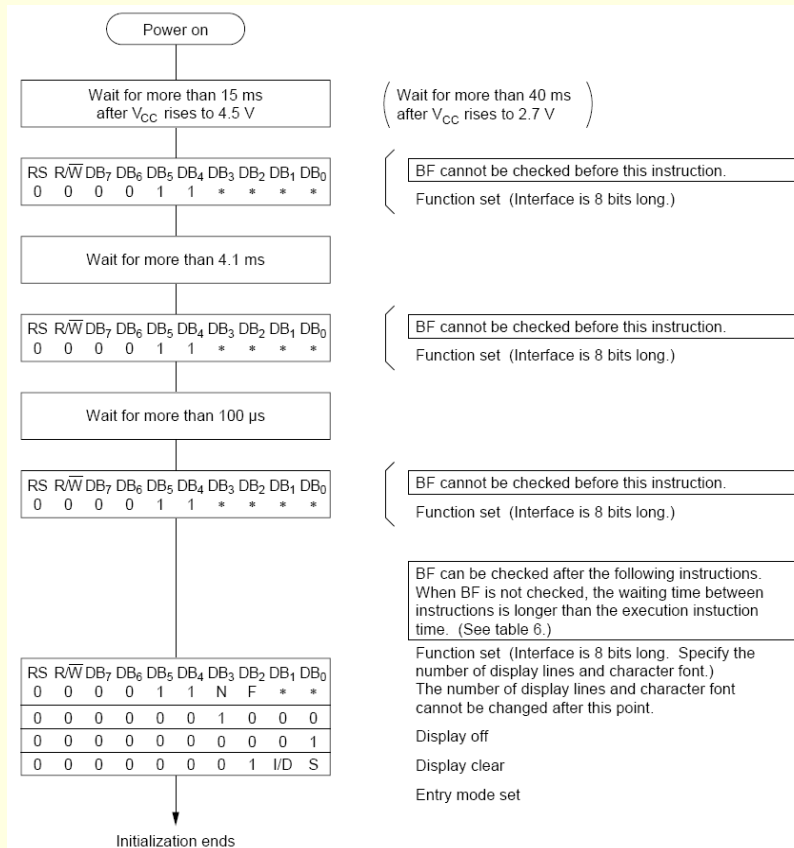


# Quelques Fonctions en C

```
■ /*****  
;Ecrit un caractère sur l'afficheur / retourne AC  
■ *****/  
  
■ unsigned char lcd_char(unsigned char caractere)  
■ {  
■ unsigned char abc;  
■ abc=waitt_bf();  
■ LCD_DATA=caractere;  
■ lcd_rs=1;  
■ pulse_e();  
■ lcd_rs=0;  
■ return(abc);  
■ }  
  
■ /*****  
; Impulsion positive sur LCD_E  
■ *****/  
■ void pulse_e(void)  
■ {  
■ lcd_e=1;  
■ lcd_e=0;  
■ }
```

```
■ /*****  
; Waitt for BF = 0 ; retourne la valeur de AC (LCD address  
Conter)  
■ *****/  
  
■ unsigned char waitt_bf(void)  
■ {  
■ unsigned char abc;  
■ LCD_DATA = 0xFF;  
■ lcd_rw=1;  
■ do {  
■     lcd_e=1;  
■     abc=LCD_DATA;  
■     lcd_e=0;  
■ }  
■ while(abc>127);  
■ abc&=0x7F;  
■ lcd_rw=0;  
■ LCD_DATA=0x00;  
■ return(abc);  
■ }
```

# Fonction d'Initialisation (8 bits)



```

void lcd_init(unsigned char function_set, unsigned char entry_mode_set)
{
    lcd_rs=0;
    lcd_rw=0;
    lcd_e=0;

    LCD_DATA=0;
    tempo(15);
    LCD_DATA=function_set;
    pulse_e();

    tempo(5);
    LCD_DATA=function_set;
    pulse_e();

    tempo(1);
    LCD_DATA=function_set;
    pulse_e();

    waitt_bf();
    LCD_DATA=function_set;
    pulse_e();

    waitt_bf();
    LCD_DATA=ALL_OFF;
    pulse_e();

    waitt_bf();
    LCD_DATA=CLEAR;
    pulse_e();

    waitt_bf();
    LCD_DATA=entry_mode_set;
    pulse_e();
}
    
```



# Quelques Déclarations en C

```
■ //CLEAR DISPLAY
■ #define CLEAR 1 //Clear DDRAM set Address at 0

■ //RETURN HOME
■ #define HOME 2 //Set DDRAM Address à 0

■ //MODE ENTRY SET
■ #define DEC_CURSOR 4 //Cursor Decrement
■ #define INC_CURSOR 6 //Cursor Increment
■ #define STOP_CURSOR 5 //Shift Entire Display

■ //DISPLAY ON/OFF
■ #define ALL_OFF 0x08 //Diplay OFF,Cursor OFF,Cursor Blink OFF
■ #define LCD_ON 0x0c //;Diplay ON
■ #define CURSOR_ON 0x0a //;Cursor ON
■ #define BLINK_ON 0x09 //;Cursor Blink ON

■ //CURSOR OR DISPLAY SHIFT
■ #define SHIFT_DISPLAY 0x18 //;Display Shift (defaut left)
■ #define SHIFT_CURSOR 0x10 //;Cursor shift (defaut left)
■ #define SHIFT_RIGHT 0x14 //;Shift righth (defaut cursor)

■ //FUNCTION SET
■ #define LCD_4BIT 0x20 //;4 bits (defaut 1 line et 5x7 dots)
■ #define LCD_8BIT 0x30 //;8 bits (defaut 1 line et 5x7 dots)
■ #define LCD_2LINE 0x28 //;2 Lines (defaut 4 bit et 5x7 dots)
■ #define LCD_50DOTS 0x24 //;5 x 10 dots (defaut 1 line et 4 bits)

■
■ extern void lcd_init(unsigned char function_set, unsigned char entry_mode_set);
■ extern void lcd_clear(void);
■ extern void lcd_home(void);
■ extern void lcd_goto(unsigned char adresse);
■ extern unsigned char lcd_function(unsigned char function);
■ extern unsigned char lcd_char(unsigned char caractere);
■ extern unsigned char lcd_string(const unsigned char *string,unsigned char adresse_lcd);
```

# Code ASCII

|   | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | A   | B   | C  | D  | E  | F   |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|----|----|----|-----|
| 0 | NUL | SOH | STH | ETH | EOT | ENQ | ACK | BEL | BS  | HT | LF  | VT  | FF | CR | SO | SI  |
| 1 | DLE | DC1 | CD2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US  |
| 2 | spc | !   | "   | #   | \$  | %   | &   | '   | (   | )  | *   | +   | ,  | -  | .  | /   |
| 3 | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | :   | ;   | <  | =  | >  | ?   |
| 4 | @   | A   | B   | C   | D   | E   | F   | G   | H   | I  | J   | K   | L  | M  | N  | O   |
| 5 | P   | Q   | R   | S   | T   | U   | V   | W   | X   | Y  | Z   | [   | \  | ]  | ^  | _   |
| 6 | `   | a   | b   | c   | d   | e   | f   | g   | h   | i  | j   | k   | l  | m  | n  | o   |
| 7 | p   | q   | r   | s   | t   | u   | v   | w   | x   | y  | z   | {   |    | }  | ~  | DEL |