TD n°4 - Mise en œuvre d'un afficheur LCD

Objectifs :

On se propose d'afficher périodiquement sur un afficheur LCD et sous forme décimale le résultat de la conversion analogique numérique du TD n°2. Les chiffres affichés occuperont les trois premiers caractères de la 1ère ligne.

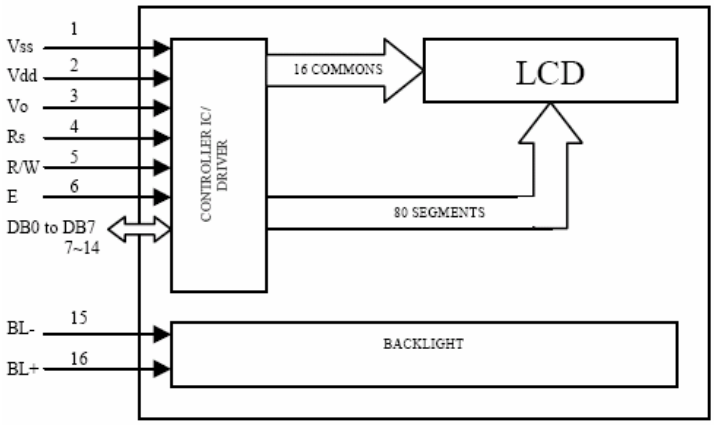
Pour faire le programme on utilisera les variables suivantes :

.def temp = r16	variable à usage général
.def tempo1 = r19	délai1
.def tempo2 = r21	délai2
.def r_conv_l = r17	résultat de la conversion pf
.def r_conv_h = r18	résultat de la conversion PF
.def cent = r15	chiffre des centaines
.def dix = r14	chiffre des dizaines
.def unit = r13	chiffre des unités
.def data = r20	donnée transmise à l'afficheur

Câblage du LCD : data sur le portC, E=bit2 portB, RS=bit0 portB et R/W à la masse.

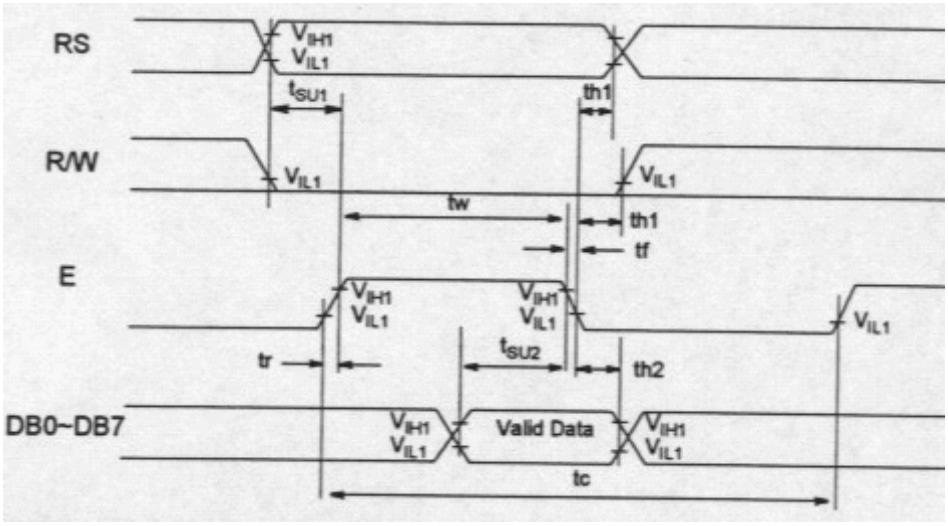
A) Transcodage/Préparation des données :
- Réaliser un algorigramme puis le sous-programme qui effectue le transcodage de (r_conv_l) => décimal puis décimal => ASCII (registres « cent », « dix » et « unit ») en vue de l'afficher sur le LCD. (ex : pour un nombre = $FF, le résultat du transcodage sera 255, pour $64 on aura 100 et ainsi de suite..). Pour transcoder un chiffre en ASCII il suffit de lui rajouter « $30 » (ex : « 5 » codé en ASCII = $35). On appellera ces programmes « **conv_bcd** » et « **conv_ascii** ».
-

B) Affichage sur le LCD :
- Réaliser les sous-programmes :
  - d'envoi de caractère (**env_car**)
  - d'envoi de donnée (**env_data**)
  - d'envoi d'instruction (**env_inst**)
  - remise à zéro de l'afficheur (**raz_lcd**)
  - d'initialisation de l'afficheur LCD en mode 8 bits, 2x16 caractères (**init_lcd**).
  - D'initialisation du curseur (**ini_curs**)
  - D'affichage des chiffres cent, dix, unit (**affiche_lcd**)



Documentation de l'afficheur :

Cycle de Lecture/écriture :



Instructions :

Instruction	Code										Description	Executed time(max) fosc=270KHz
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Clear all display and returns the cursor to the home position (Address 0)	1.53 ms
Cursor at home	0	0	0	0	0	0	0	0	1	*	Returns the cursor to the home position (Address 0). Also return the display being shifted to the original position. DDRAM contents remain unchanged.	1.53 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	Sets the cursor move direction and specifies or not to shift the display. These operations are performed during data write and read.	39 µs
Display on/off control	0	0	0	0	0	0	1	D	C	B	Sets the ON/OFF of all display (D) cursor ON/OFF (C), and blink of cursor position character (B).	39 µs
Cursor/display shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves the cursor and shifts the display without changing the DDRAM contents.	39 µs
Function set	0	0	0	0	1	DL	N	F	*	*	Sets interface data length (DL), number of display lines (N) and character font (F).	39 µs
CGRAM address set	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Sets the CGRAM, data is sent and received after this setting.	39 µs
DDRAM address set	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set the DGRAM, data is sent and received after this setting.	39 µs
Busy Flag/ address read	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Recall Busy flag (FB) indicating internal operation is being performed and read address counter contents.	0 µs
CGRAM/DDRAM data write	1	0	WRITE DATA								Writes data into DDRAM or CGRAM	43 µs
CGRAM/DDRAM data read	1	1	READ DATA								Reads data into DDRAM or CGRAM	43 µs

Code		Description	Executed time (max)
I/D=1: Increment	DL= 0:4-bit	DDRAM: Display Data RAM	Fcp or fosc = 250KHz
I/D=0: Decrement	1/16 duty	CGRAM: Character Generator RAM	However, when frequency changes, execution
S=1: With display shift	1/8 duty, 1/11 duty	ACG: CGRAM Address	time also changes.
S/C=1: Display shift	F= 1:5x10 dots	ADD: DDRAM Address	
S/C=0: Cursor movement	F= 0: 5x7 dots	Corresponds to cursor address	Example
R/L=1: Shift to the right	BF=1: Internal operations is	AC: Address Counter, used for both DDRAM	If fcp or fosc is 270KHz
R/L=0: Shift to the left	being performed	and CGRAM	
DL=1: 8-bit	BF=0: Instruction acceptable	* : Invalid	

Exemple d'initialisation :

8 bit operation, 8 digit 2line display example.

Step	Instruction											
No	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Display	Operation
1	Power supply on											Initialized. No display
2	Function set											Sets to 8 bit operation and select 2 lines display and 5x8 dot character font.
	0	0	0	0	1	1	1	0	X	X		
3	Display on/off control										_	Turn on display and cursor. All display is in space mode because of initialisation.
	0	0	0	0	0	0	1	1	1	0		
4	Entry mode set										_	Sets mode to increment the address by one and to shift the cursor to the right at the time of write to the DD/CGRAM. Display is not shifted.
	0	0	0	0	0	0	0	1	1	0		
5	Write data to CGRAM/DDRAM										C_	Writes C. DDRAM has already been selected by initialisation when the power was turned on. The cursor is incremented by one and shifted to the right.
	1	0	0	1	0	0	0	0	1	1		
6												
7	Write data to CGRAM/DDRAM										CRYSTAL CLEAR_	Writes R
	1	0	0	1	0	1	0	0	1	0		
8	Set DDRAM address										CRYSTAL CLEAR _	Sets DDRAM address so that the cursor is positioned at the head of the second line.
	0	0	1	1	0	0	0	0	0	0		
9	Write data to CGRAM/DDRAM										CRYSTAL CLEAR T_	Writes T
	1	0	0	1	0	1	0	1	0	0		
10												
11	Write data to CGRAM/DDRAM										CRYSTAL CLEAR TECH_	Writes H
	1	0	0	1	0	0	1	0	0	0		
12	Entry mode set										CRYSTAL CLEAR TECH_	Sets mode to shift display at the time of write.
	0	0	0	1	0	0	0	1	1	1		
13	Write data to CGRAM/DDRAM										CRYSTAL CLEAR TECHNOLOGY	Writes Y. Display is shifted to the left. The first and second lines both shift at the same time.
	1	0	0	1	0	1	1	0	0	1		
14												
15	Return home										CRYSTAL CLEAR TECHNOLOGY	Returns both display and cursor to the original position (address 0)
	0	0	0	1	0	0	0	0	1	X		

Programmes correspondants :

```
;---------------- conv_bcd : conversion binaire => décimal--------
conv_bcd:
        clr cent
        clr dix
        clr unit
conv_bcd_0:
        cpi r_conv_l, $64
        brcs conv_bcd_1
        inc cent
        subi r_conv_l,$64
        rjmp conv_bcd_0
conv_bcd_1:
        cpi r_conv_l, $A
        brcs conv_bcd_2
        inc dix
        subi r_conv_l,$A
        rjmp conv_bcd_1
conv_bcd_2:
        mov unit, r_conv_l
        ret
;-------------------------------------------------------------


;------------------ conv_ascii : conversion ASCII----------------
conv_ascii:
        ldi temp, $30
        add cent, temp
        add dix,  temp
        add unit, temp
        ret
;-------------------------------------------------------------


;----------------------- affiche_lcd: affiche sur LCD--------------
;
; les chiffres binaires dans les registres cent, dix, unit
; sont affichés sur les 3 premiers digits du LCD
;   câblage: data sur le portC, E=bit2 portB, RS=bit0 portB
; et R/W à la masse
;-------------------------------------------------------------
affiche_lcd:
        rcall ini_curs   ; curseur en haut à gauche
        mov data, cent
        rcall env_data ; affichage du contenu de cent
        mov data,dix
        rcall env_data ; affichage du contenu de dix
        mov data,unit
        rcall env_data ; affichage du contenu de unit
        ret                                ; retour au prog appelant
;-------------------------------------------------------------
```

```
;------------ env_inst: positionne RS pour envoi instruction----------
env_inst:
        cbi portb,0             ; mise à 0 de RS (envoi instruction)
        rcall env_car
        ret
;-------------------------------------------------------------

;------------- env_data: positionne RS pour envoi donnée---------------
env_data:
        sbi portb,0             ; mise à 1 de RS (envoi donnée)
        rcall env_car
        ret
;-------------------------------------------------------------

;------------ env_car:  positionne E pour envoi d'un caractère--------
env_car:
        out portc,data  ; positionne data sur afficheur
        rcall delai1
        sbi portb,2             ;mise à 1 de E
        rcall delai1
        cbi portb,2             ; mise à 0 de E
        ret
;-------------------------------------------------------------

;------- init_lcd:  initialise le LCD----------------------------
init_lcd:

        clr data
        out portb,data
        rcall delai1
        ldi data, 0x38  ; mode 8 bits,2 lignes
        rcall env_inst
        ldi data,0x0E   ;dislay et cursor on
        rcall env_inst
        ldi data,0x06   ;mode increment droite
        rcall env_inst
        rcall raz_lcd
        ret
;-------------------------------------------------

;------------- ini_curs:  initialise le curseur sur 1er digit------
ini_curs:
        ldi data, 0x02
        rcall env_inst
        ret
;-------------------------------------------------
;-------------- raz_lcd:  remet à 0 le LCD----------------------
raz_lcd:
        ldi data,0x01
        rcall env_inst
        ret
;-------------------------------------------------
```