07/03/2023

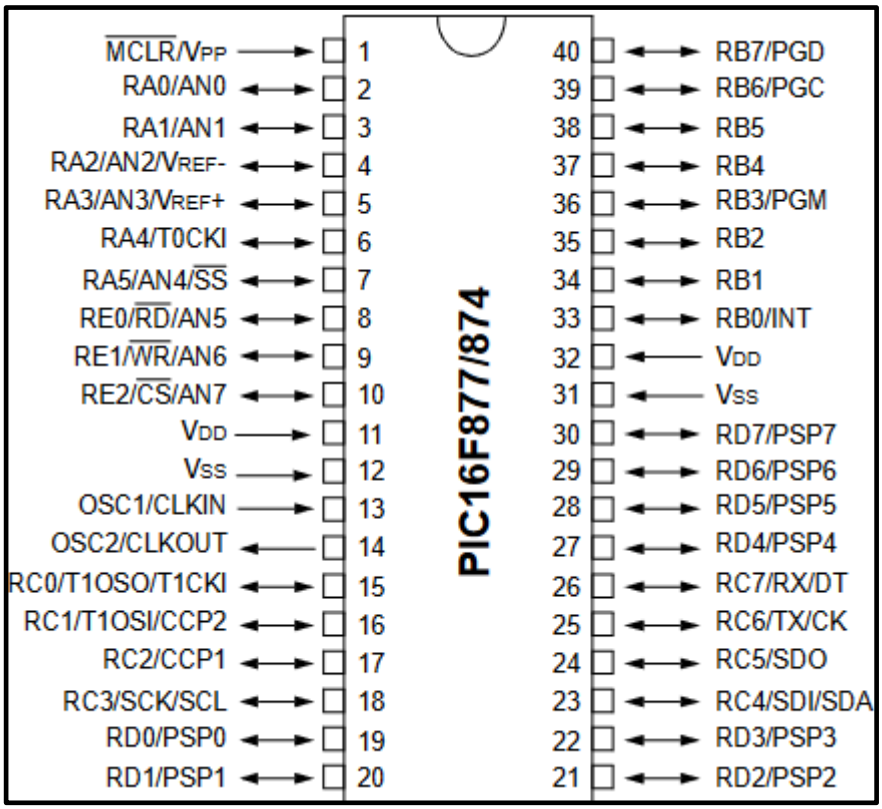
LE MICROCONTROLEUR PIC16F877 et Ecran LCD

Principales données

Abdussalam Gemal
ESME SUDRIA

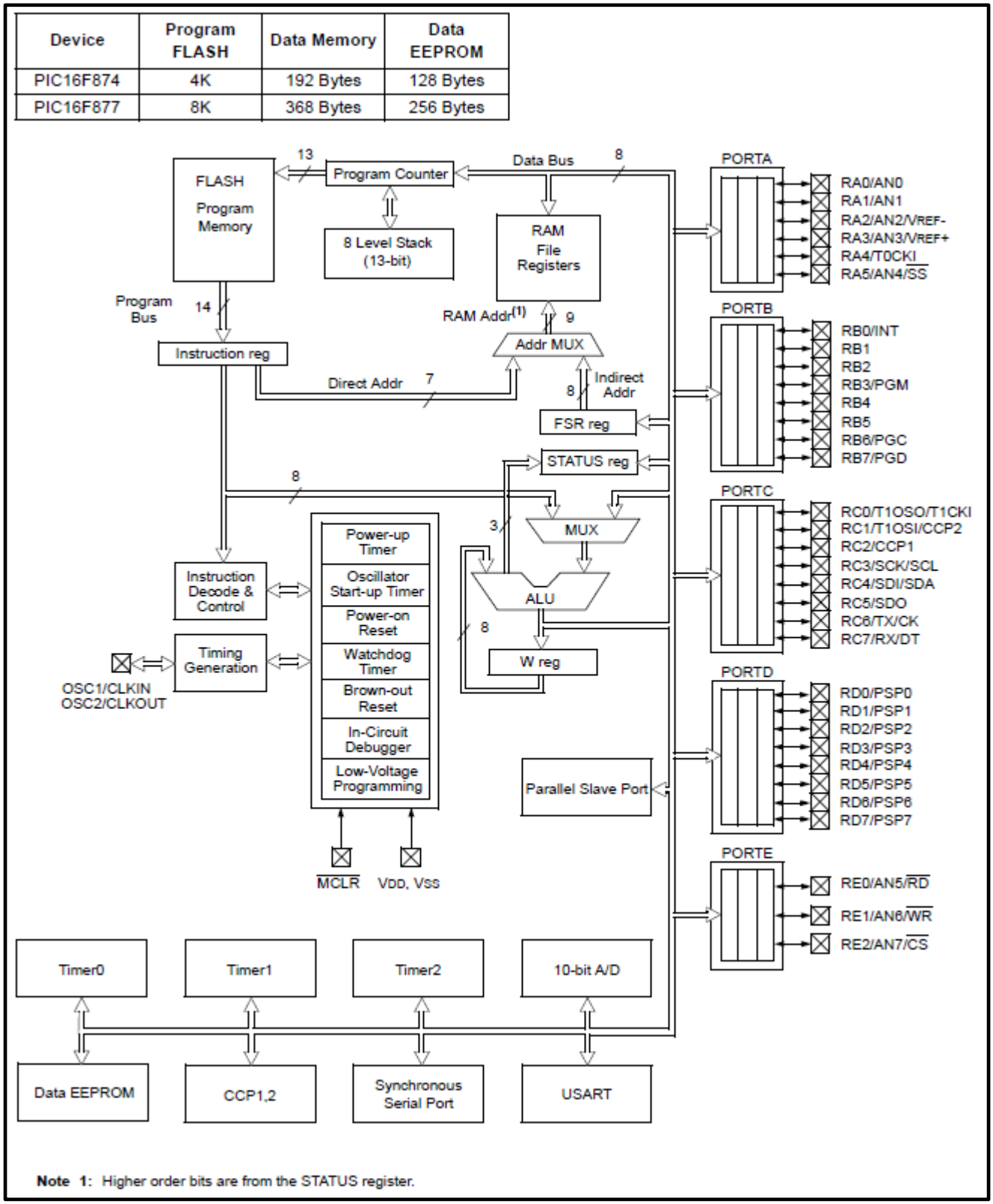
LE MICROCONTROLEUR

I) Le brochage



II)Architecture interne

Device	Program FLASH	Data Memory	Data EEPROM
PIC16F874	4K	192 Bytes	128 Bytes
PIC16F877	8K	368 Bytes	256 Bytes



PIC16F877

III) Organisation de la mémoire de programme

FIGURE 2-1: PIC16F877/876 PROGRAM MEMORY MAP AND STACK

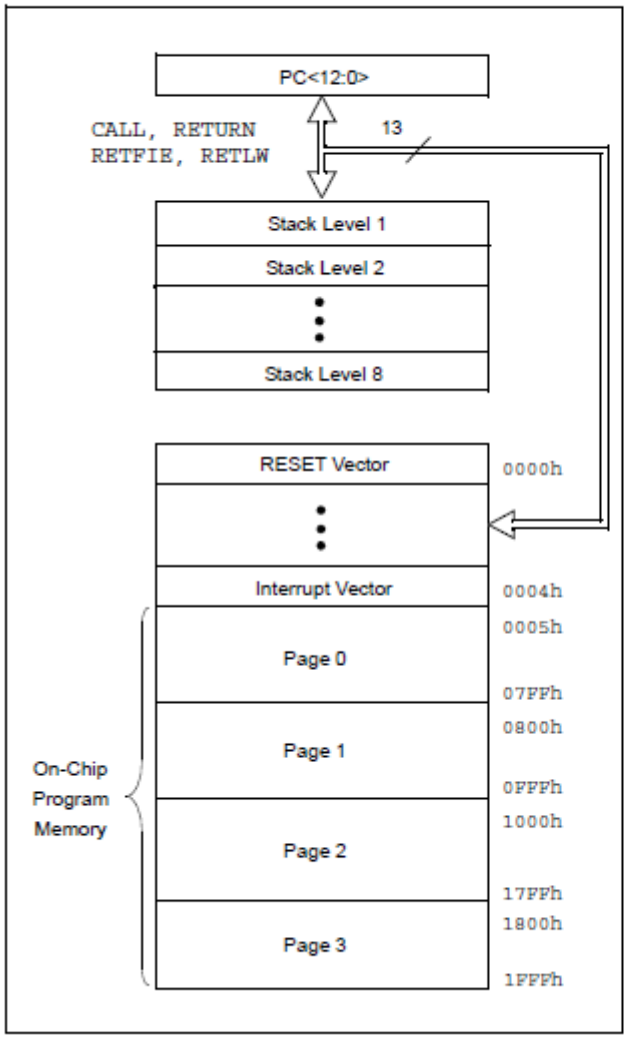
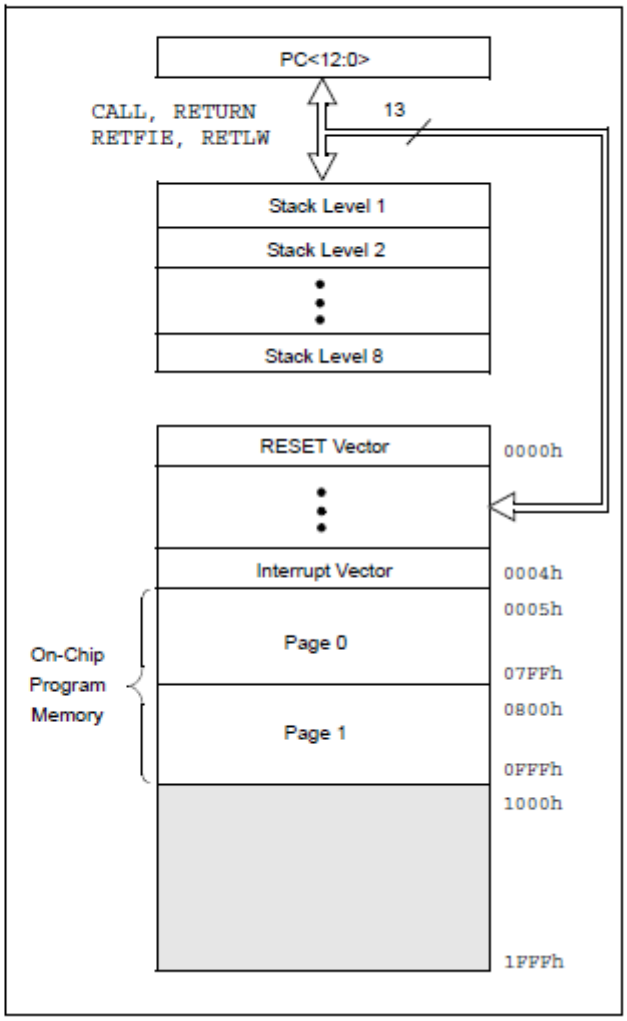


FIGURE 2-2: PIC16F874/873 PROGRAM MEMORY MAP AND STACK



IV) Organisation de la mémoire de données

2.2 Data Memory Organization

The data memory is partitioned into multiple banks which contain the General Purpose Registers and the Special Function Registers. Bits RP1 (STATUS<6>) and RP0 (STATUS<5>) are the bank select bits.

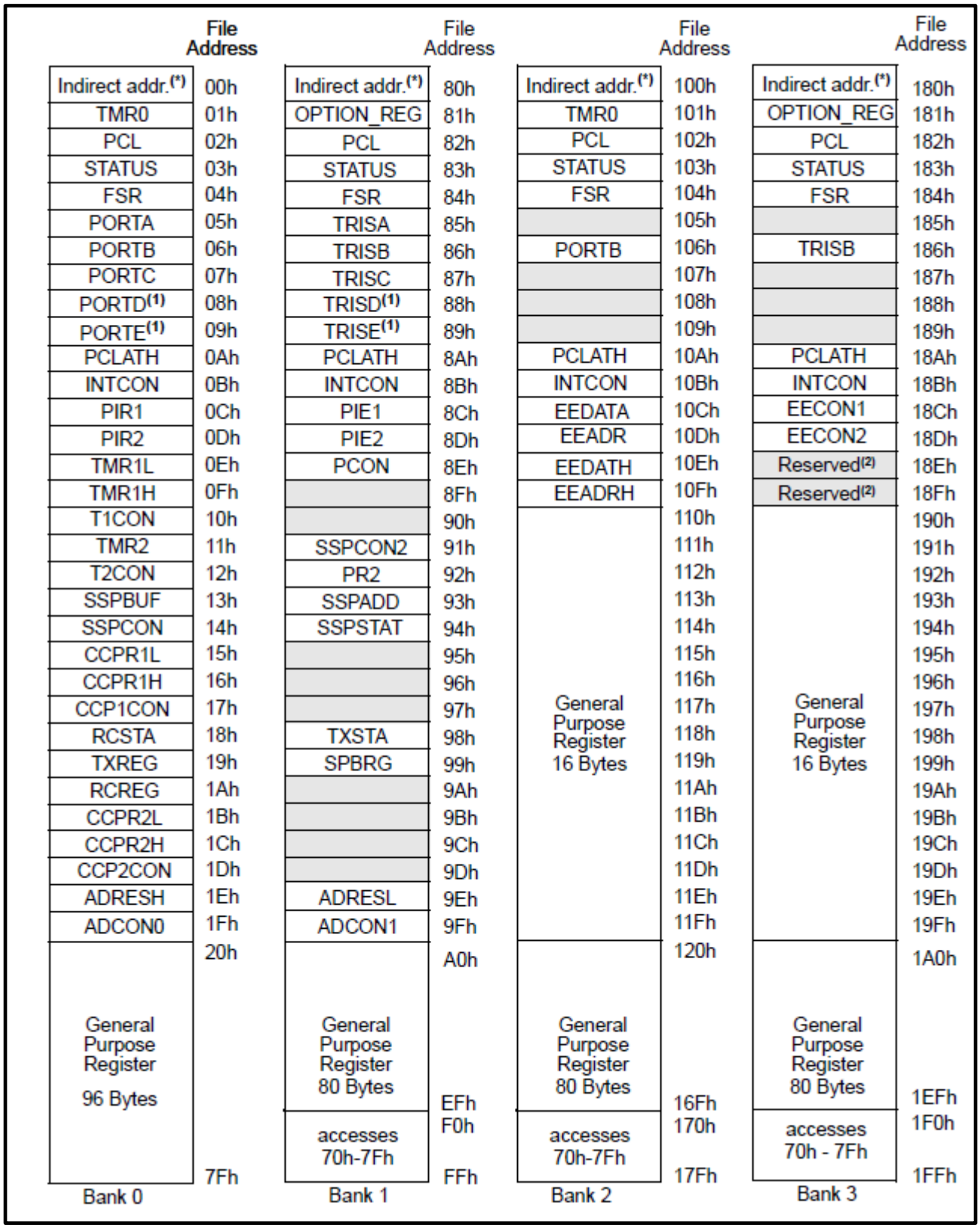
RP1:RP0	Bank
00	0
01	1
10	2
11	3

Each bank extends up to 7Fh (128 bytes). The lower locations of each bank are reserved for the Special Function Registers. Above the Special Function Registers are General Purpose Registers, implemented as static RAM. All implemented banks contain Special Function Registers. Some frequently used Special Function Registers from one bank may be mirrored in another bank for code reduction and quicker access.

Note: EEPROM Data Memory description can be found in Section 4.0 of this data sheet.

2.2.1 GENERAL PURPOSE REGISTER FILE

The register file can be accessed either directly, or indirectly through the File Select Register (FSR).


File Address
Indirect addr.(*) 00h
TMR0 01h
PCL 02h
STATUS 03h
FSR 04h
PORTA 05h
PORTB 06h
PORTC 07h
PORTD(1) 08h
PORTE(1) 09h
PCLATH 0Ah
INTCON 0Bh
PIR1 0Ch
PIR2 0Dh
TMR1L 0Eh
TMR1H 0Fh
T1CON 10h
TMR2 11h
T2CON 12h
SSPBUF 13h
SSPCON 14h
CCPR1L 15h
CCPR1H 16h
CCP1CON 17h
RCSTA 18h
TXREG 19h
RCREG 1Ah
CCPR2L 1Bh
CCPR2H 1Ch
CCP2CON 1Dh
ADRESH 1Eh
ADCON0 1Fh
20h
General Purpose Register
96 Bytes
7Fh
Bank 0
File Address
Indirect addr.(*) 80h
OPTION_REG 81h
PCL 82h
STATUS 83h
FSR 84h
TRISA 85h
TRISB 86h
TRISC 87h
TRISD(1) 88h
TRISE(1) 89h
PCLATH 8Ah
INTCON 8Bh
PIE1 8Ch
PIE2 8Dh
PCON 8Eh
8Fh
90h
SSPCON2 91h
PR2 92h
SSPADD 93h
SSPSTAT 94h
95h
96h
97h
TXSTA 98h
SPBRG 99h
9Ah
9Bh
9Ch
9Dh
ADRESL 9Eh
ADCON1 9Fh
A0h
General Purpose Register
80 Bytes
EFh
F0h
accesses 70h-7Fh
FFh
Bank 1
File Address
Indirect addr.(*) 100h
TMR0 101h
PCL 102h
STATUS 103h
FSR 104h
105h
PORTB 106h
107h
108h
109h
PCLATH 10Ah
INTCON 10Bh
EEDATA 10Ch
EEADR 10Dh
EEDATH 10Eh
EEADRH 10Fh
110h
111h
112h
113h
114h
115h
116h
117h
118h
119h
11Ah
11Bh
11Ch
11Dh
11Eh
11Fh
120h
General Purpose Register
16 Bytes
General Purpose Register
80 Bytes
16Fh
170h
accesses 70h-7Fh
17Fh
Bank 2
File Address
Indirect addr.(*) 180h
OPTION_REG 181h
PCL 182h
STATUS 183h
FSR 184h
185h
TRISB 186h
187h
188h
189h
PCLATH 18Ah
INTCON 18Bh
EECON1 18Ch
EECON2 18Dh
Reserved(2) 18Eh
Reserved(2) 18Fh
190h
191h
192h
193h
194h
195h
196h
197h
198h
199h
19Ah
19Bh
19Ch
19Dh
19Eh
19Fh
1A0h
General Purpose Register
16 Bytes
General Purpose Register
80 Bytes
1EFh
1F0h
accesses 70h - 7Fh
1FFh
Bank 3

PIC16F877

V) Le registre d'état : Registre STATUS

2.2.2.1 STATUS Register

The STATUS register contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory.

The STATUS register can be the destination for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{TO}$ and $\overline{PD}$ bits are not writable, therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as `000u uluu` (where `u` = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect the Z, C or DC bits from the STATUS register. For other instructions not affecting any status bits, see the "Instruction Set Summary."

Note: The C and DC bits operate as a $\overline{\text{borrow}}$ and $\overline{\text{digit borrow}}$ bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

REGISTER 2-1: STATUS REGISTER (ADDRESS 03h, 83h, 103h, 183h)

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP:** Register Bank Select bit (used for indirect addressing)
1 = Bank 2, 3 (100h - 1FFh)
0 = Bank 0, 1 (00h - FFh)

bit 6-5 **RP1:RP0:** Register Bank Select bits (used for direct addressing)
11 = Bank 3 (180h - 1FFh)
10 = Bank 2 (100h - 17Fh)
01 = Bank 1 (80h - FFh)
00 = Bank 0 (00h - 7Fh)
Each bank is 128 bytes

bit 4 $\overline{TO}$: Time-out bit
1 = After power-up, `CLRWDT` instruction, or `SLEEP` instruction
0 = A WDT time-out occurred

bit 3 $\overline{PD}$: Power-down bit
1 = After power-up or by the `CLRWDT` instruction
0 = By execution of the `SLEEP` instruction

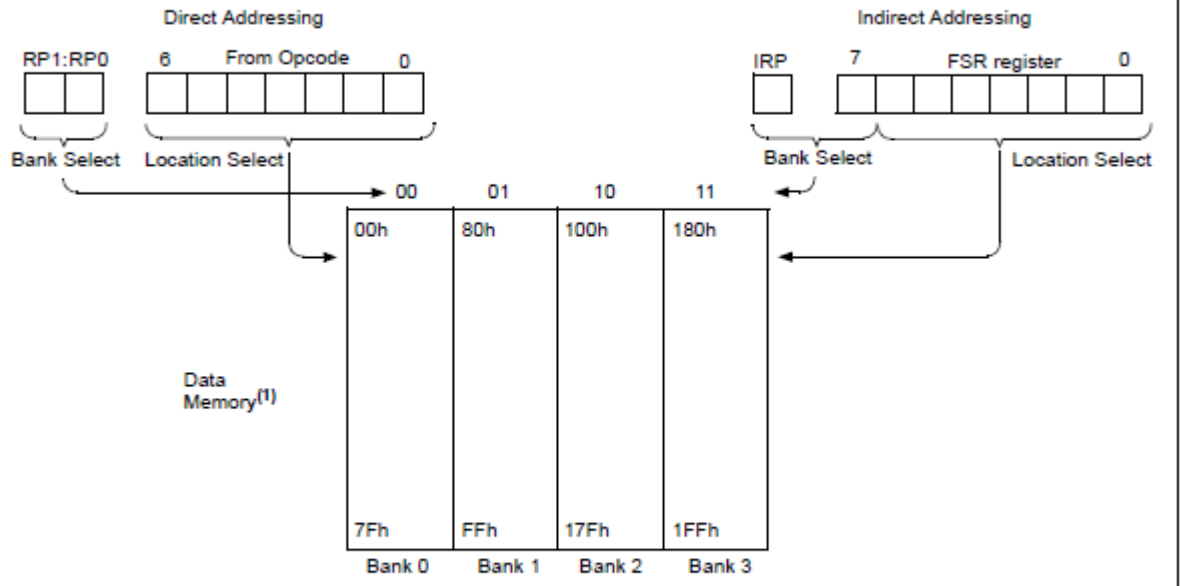
bit 2 Z: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 DC: Digit carry/$\overline{\text{borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
(for $\overline{\text{borrow}}$, the polarity is reversed)
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0 C: Carry/$\overline{\text{borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

VI) Accès aux données en direct ou indirect

FIGURE 2-6: DIRECT/INDIRECT ADDRESSING



Note 1: For register file map detail, see Figure 2-3.

VII) Les registres PORTX et TRISX

3.3 PORTC and the TRISC Register

PORTC is an 8-bit wide, bi-directional port. The corresponding data direction register is TRISC. Setting a TRISC bit (= 1) will make the corresponding PORTC pin an input (i.e., put the corresponding output driver in a Hi-Impedance mode). Clearing a TRISC bit (= 0) will make the corresponding PORTC pin an output (i.e., put the contents of the output latch on the selected pin).

PORTC is multiplexed with several peripheral functions (Table 3-5). PORTC pins have Schmitt Trigger input buffers.

VIII) Le convertisseur analogique numérique

11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the other devices.

The analog input charges a sample and hold capacitor. The output of the sample and hold capacitor is the input into the converter. The converter then generates a digital result of this analog level via successive approximation. The A/D conversion of the analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low voltage reference input that is software selectable to some combination of VDD, VSS, RA2, or RA3.

The A/D converter has a unique feature of being able to operate while the device is in SLEEP mode. To operate in SLEEP, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register0 (ADCON0)
- A/D Control Register1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference), or as digital I/O.

Additional information on using the A/D module can be found in the PIC® MCU Mid-Range Family Reference Manual (DS33023).

REGISTER 11-1: ADCON0 REGISTER (ADDRESS: 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits
00 = FOSC/2
01 = FOSC/8
10 = FOSC/32
11 = FRC (clock derived from the internal A/D module RC oscillator)

bit 5-3 **CHS2:CHS0**: Analog Channel Select bits
000 = channel 0, (RA0/AN0)
001 = channel 1, (RA1/AN1)
010 = channel 2, (RA2/AN2)
011 = channel 3, (RA3/AN3)
100 = channel 4, (RA5/AN4)
101 = channel 5, (RE0/AN5)(1)
110 = channel 6, (RE1/AN6)(1)
111 = channel 7, (RE2/AN7)(1)

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
If ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion)
0 = A/D conversion not in progress (this bit is automatically cleared by hardware when the A/D conversion is complete)

bit 1 **Unimplemented**: Read as '0'

bit 0 **ADON**: A/D On bit
1 = A/D converter module is operating
0 = A/D converter module is shut-off and consumes no operating current

Note 1: These channels are not available on PIC16F873/876 devices.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 11-2: ADCON1 REGISTER (ADDRESS 9Fh)

U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. 6 Most Significant bits of ADRESH are read as '0'.
0 = Left justified. 6 Least Significant bits of ADRESL are read as '0'.

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0**: A/D Port Configuration Control bits:

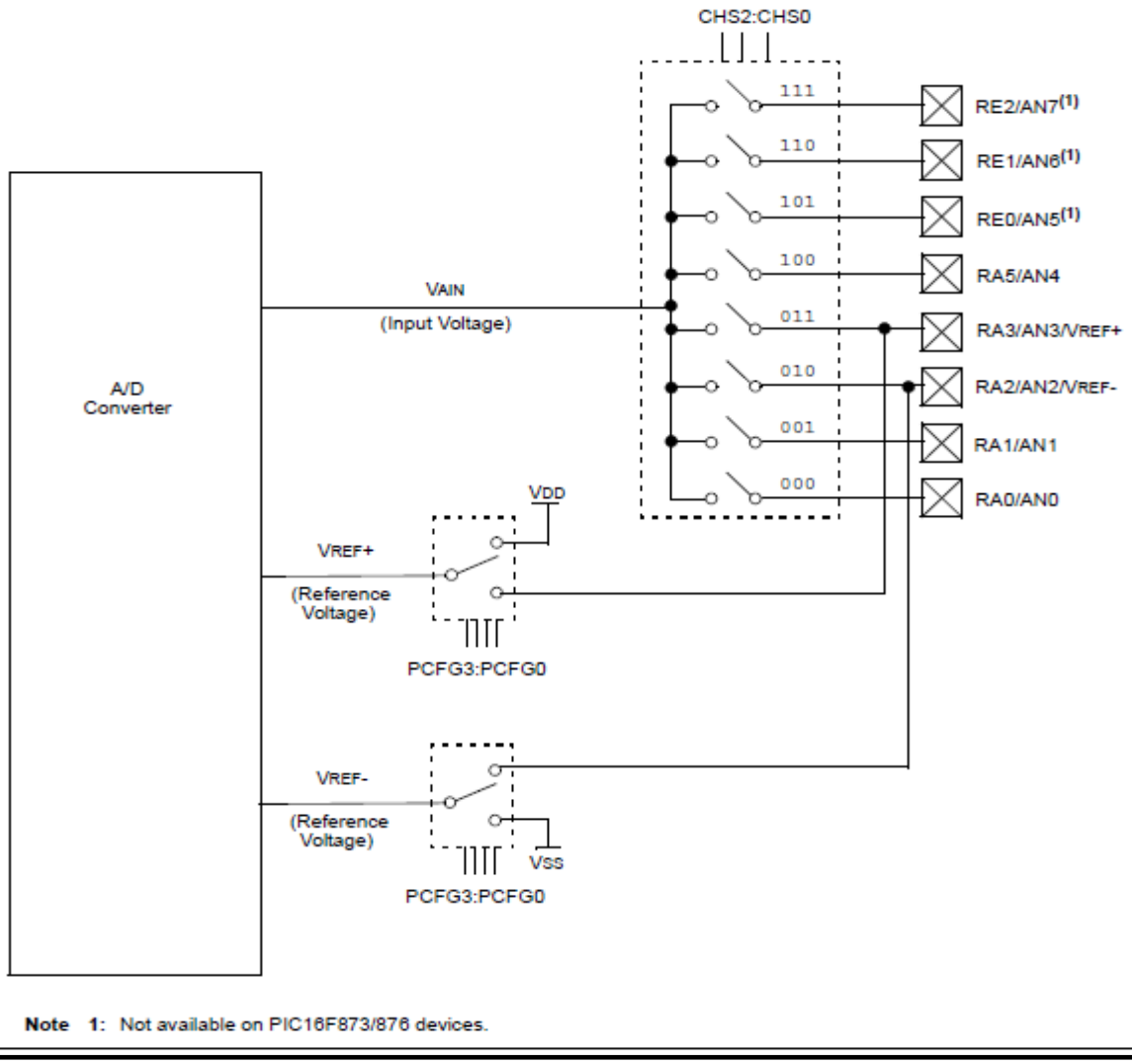
PCFG3: PCFG0	AN7(1) RE2	AN6(1) RE1	AN5(1) RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	VREF+	VREF-	CHAN/ Refs(2)
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	RA3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	RA3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	RA3	VSS	2/1
011x	D	D	D	D	D	D	D	D	VDD	VSS	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	RA3	RA2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	RA3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	RA3	RA2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	RA3	RA2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	RA3	RA2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	RA3	RA2	1/2

A = Analog input D = Digital I/O

Note 1: These channels are not available on PIC16F873/876 devices.
2: This column indicates the number of analog channels available as A/D inputs and the number of analog channels used as voltage reference inputs.

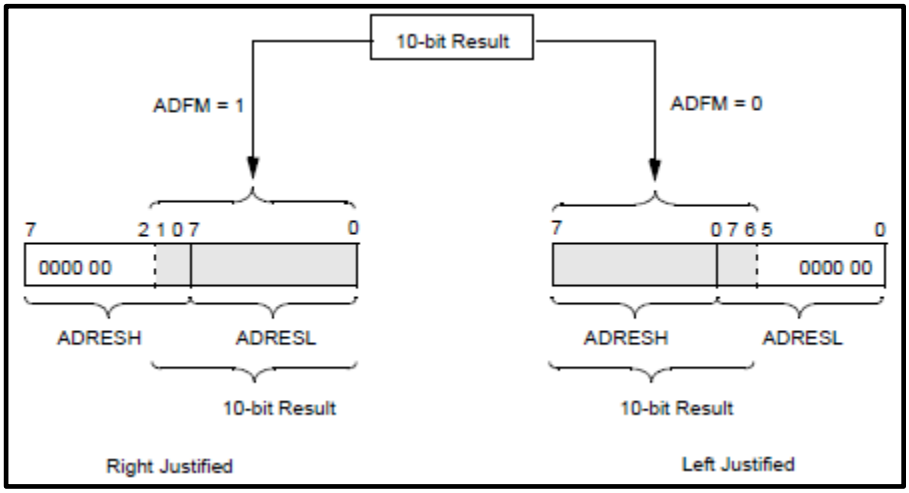
Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

FIGURE 11-1: A/D BLOCK DIAGRAM



Note 1: Not available on PIC16F873/876 devices.

AD Clock Source (TAD)		Maximum Device Frequency
Operation	ADCS1:ADCS0	Max.
2TOSC	00	1.25 MHz
8TOSC	01	5 MHz
32TOSC	10	20 MHz
RC(1, 2, 3)	11	(Note 1)



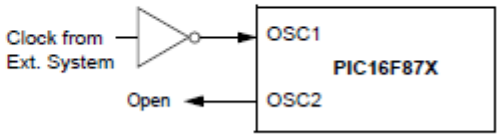
IX) Les sources d'horloge

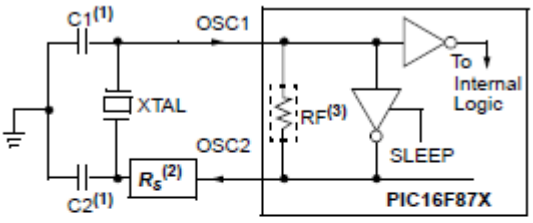
12.2.1 OSCILLATOR TYPES

The PIC16F87X can be operated in four different oscillator modes. The user can program two configuration bits (FOSC1 and FOSC0) to select one of these four modes:

- LP Low Power Crystal
- XT Crystal/Resonator
- HS High Speed Crystal/Resonator
- RC Resistor/Capacitor

OPERATION (HS, XT OR LP OSC CONFIGURATION)



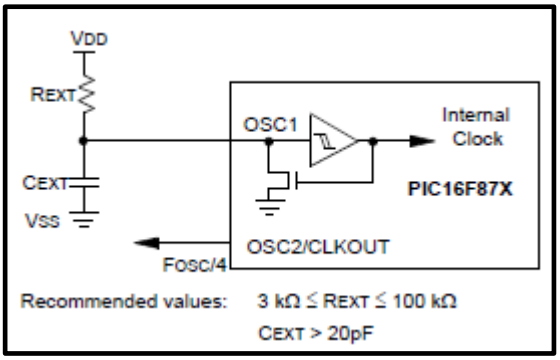


Note 1: See Table 12-1 and Table 12-2 for recommended values of C1 and C2.

2: A series resistor (R_S) may be required for AT strip cut crystals.

3: RF varies with the crystal chosen.

Ranges Tested:			
Mode	**Freq.**	**OSC1**	**OSC2**
XT	455 kHz 2.0 MHz 4.0 MHz	68 - 100 pF 15 - 68 pF 15 - 68 pF	68 - 100 pF 15 - 68 pF 15 - 68 pF
HS	8.0 MHz 16.0 MHz	10 - 68 pF 10 - 22 pF	10 - 68 pF 10 - 22 pF
These values are for design guidance only. See notes following Table 12-2.			
Resonators Used:			
455 kHz	Panasonic EFO-A455K04B	± 0.3%	
2.0 MHz	Murata Erie CSA2.00MG	± 0.5%	
4.0 MHz	Murata Erie CSA4.00MG	± 0.5%	
8.0 MHz	Murata Erie CSA8.00MT	± 0.5%	
16.0 MHz	Murata Erie CSA16.00MX	± 0.5%	
All resonators used did not have built-in capacitors.			

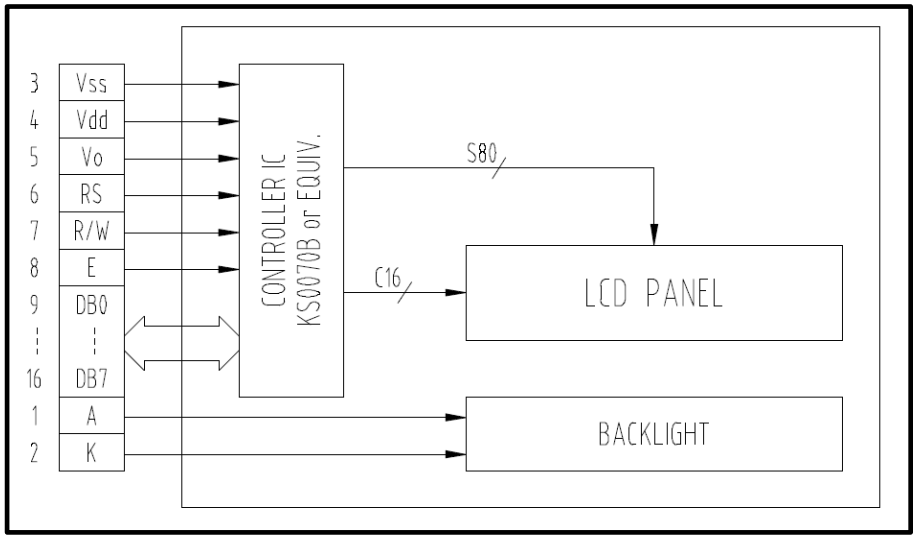


X) Les instructions

Mnémonique		Description	Nbre de cycles	Flag(s) affecté(s)
Les instructions orientées octet				
ADDWF	f,d	(W) + (f) → (destination)	1	C,DC,Z
ANDWF	f,d	(W) .AND. (f) → (W)	1	Z
CLRF	f	0 → (f)	1	Z
CLRW		0 → (W)	1	Z
COMF	f,d	/(f) → (destination)	1	Z
DECF	f,d	(f) - 1 → (destination)	1	Z
DECFSZ	f,d	(f) - 1 → (destination). Saute l'instruction suivante si résultat = 0	1 ou 2	
INCF	f,d	(f) + 1 → (destination)	1	Z
INCFSZ	f,d	(f) + 1 → (destination). Saute l'instruction suivante si résultat = 0	1 ou 2	
IORWF	f,d	(W) .OR. (f) → (W)	1	Z
MOVF	f,d	(f) → (destination)	1	Z
MOVWF	f,d	(W) → (f)	1	
NOP		Pas d'opération	1	
RLF	f,d	Rotation à gauche	1	C
RRF	f,d	Rotation à droite	1	C
SUBWF	f,d	(f) - (W) → (destination)	1	C,DC,Z
SWAPF	f,d	(f<3:0>)→(destination<7:4>),(f<7:4>)→(destination<3:0>)	1	
XORWF	f, d	(W) .XOR. (f) → (destination)	1	Z
Les instructions orientées bit				
BCF	f,b	Mettre le bit n° b de f à 0	1	
BSF	f,b	Mettre le bit n° b de f à 1	1	
BTFSC	f,b	Tester le bit n° b de f. Saute l'instruction suivante si b=0	1 ou 2	
BTFSS	f,b	Tester le bit n° b de. Saute l'instruction suivante si b=1	1 ou 2	
Les instructions littérales et de contrôle				
ADDLW	K	(W) + k → (W). K est une valeur littérale entre 0 et 255	1	C,DC,Z
ANDLW	K	(W) .AND. k → (W).	1	Z
CALL	K	Appel d'une fonction stockée à partir de l'étiquette K	2	
GOTO	K	Saut à l'étiquette K (adresse K)	2	
MOVLW	K	Charger K dans W	1	
RETURN		Retour au programme principal (une fois CALL terminé)	2	
SUBLW	K	k - (W) → (W)	1	C,DC,Z
IORLW	K	(W) .OR. k → (W).		
RETFIE		Fin du programme d'interruption		
RETLW	K	Fin du programme d'interr. et chargement de K dans (W)		
XORLW	K	(W) .XOR. k → (W).		
SLEEP		Mode veille		

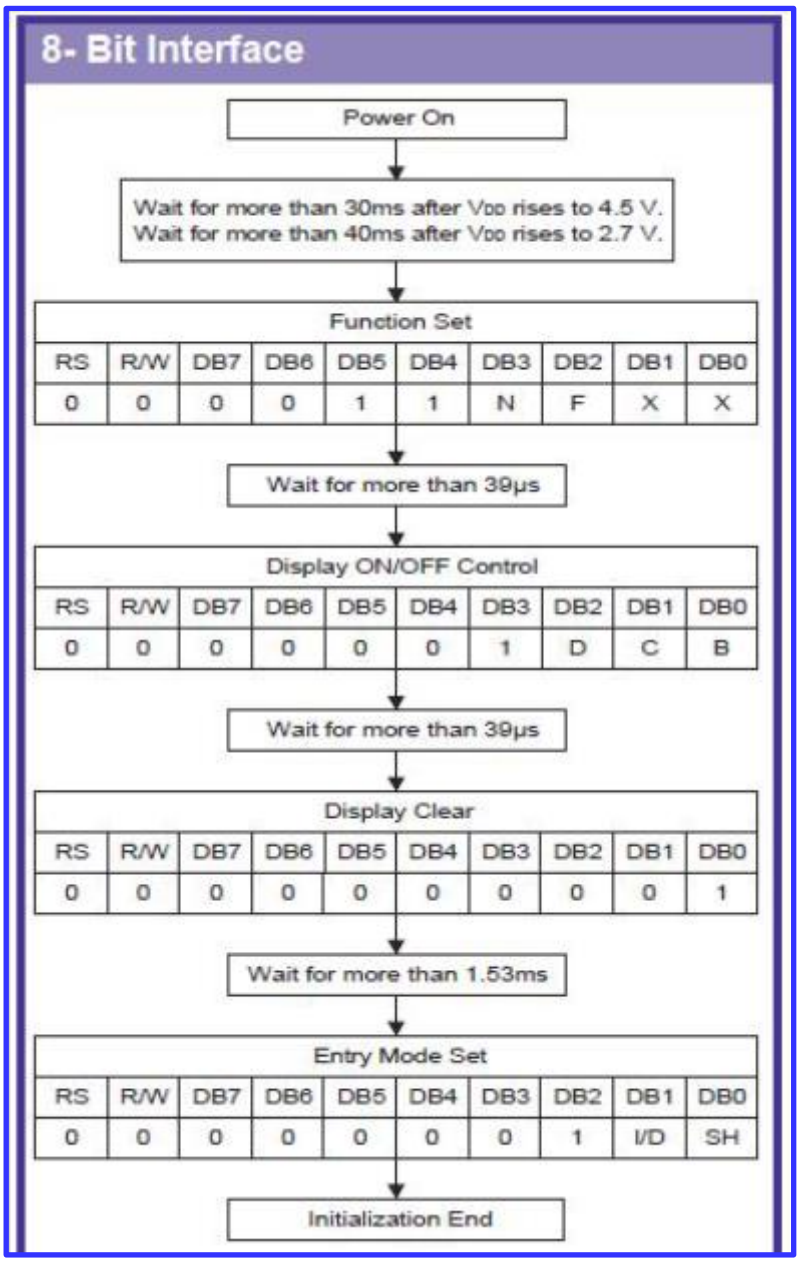
LE LCD

I) Le brochage et affectation des broches



Pin NO.	Symbol	Level	Description
1	A	---	Backlight anode
2	K	---	Backlight cathode
3	VSS	0V	Ground
4	VDD	5.0V	Supply voltage for logic
5	VO	---	Input voltage for LCD
6	RS	H/L	H : Data signal, L : Instruction signal
7	R/W	H/L	H : Read mode, L : Write mode
8	E	H, H → L	Chip enable signal
9	DB0	H/L	Data bit 0
10	DB1	H/L	Data bit 1
11	DB2	H/L	Data bit 2
12	DB3	H/L	Data bit 3
13	DB4	H/L	Data bit 4
14	DB5	H/L	Data bit 5
15	DB6	H/L	Data bit 6
16	DB7	H/L	Data bit 7

II) Procédure d'initialisation



D/L	0	4-bit mode
	1	8-bit mode
D	0	Display off
	1	Display on
I/D	0	Decrement mode
	1	Increment mode
N	0	1-line mode
	1	2-line mode
C	0	Cursor off
	1	Cursor on
SH	0	Entire shift off
	1	Entire shift on
F	0	Display off
	1	Display on
B	0	Blink off
	1	Blink on

III) Liste des Commandes

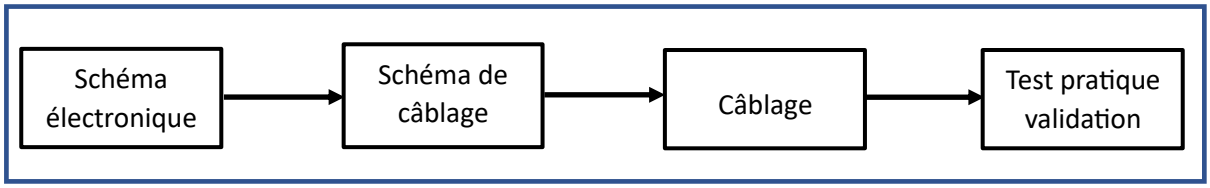
Instruction	Instruction Code										Description Instruction Code	Execution time (fsoc=270kHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM. and set DDRAM address to "00H" from AC.	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	X	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and make shift of entire display enable.	39µs
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	39µs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	X	X	Set cursor moving and display shift control bit, and the direction, without changing DDRAM data.	39µs
Function Set	0	0	0	0	1	DL	N	F	X	X	Set interface data length (DL : 4-bit/8-bit), numbers of display line (N : 1-line/2-line), display font type(F : 5 X 8 dots/ 5 X 11 dots)	39µs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39µs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39µs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0µs
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43µs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43µs

NOTE: When an MPU program with checking the Busy Flag (DB7) is made, it must be necessary 1/2 fosc is necessary for executing the next instruction by the falling edge of the 'E' signal after the Busy Flag (DB7) goes to "LOW".

Characters	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
First line	00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
Second line	40H	41H	42H	43H	44H	45H	46H	47H	48H	49H	4AH	4BH	4CH	4DH	4EH	4FH

LA PARTIE PRATIQUE

Toutes les fonctions doivent être réalisées suivant les indications suivantes :



1) **Première étape : L'alimentation électrique**
   a) Schéma électronique
   b) Simulation
   c) Schéma de câblage
   d) Câblage
   e) Vérification du fonctionnement

2) **Deuxième étape : Le µC et les circuits associés (Quartz, RESET et Connecteur du PICKIT)**
   a) Schéma électronique
   b) Schéma de câblage
   c) Faire l'organigramme de génération d'un signal carré sur une broche ;
   d) Traduire l'organigramme ci-dessus en langage assembleur (le fichier est disponible sous Teams : Sequelette_Programme.asm);
   e) Simulation avec MPLAB IDE et visualisation avec l'analyseur logique du signal carré ;
   f) Câblage
   g) Programmation du PIC et visualisation du signal carré avec un oscilloscope ;
   h) Sauvegarder le programme sous le nom **signal_carré.asm**.

3) **Troisième étape : le CAN :** voir le support disponible sur Teams
   a) Faire le schéma électronique avec un potentiomètre câblé sur l'entrée du PORTA configurée en entrée analogique ;
   b) Faire le schéma de câblage ;
   c) Faire l'organigramme du CAN ;
   d) Traduire l'organigramme ci-dessus en langage assembleur ;
   e) Programmer le PIC puis vérifier, grâce à l'analyseur logique, sur la carte que les 8 bits du PORT sélectionné pour le bus de donnée, évoluent proportionnellement à la tension fournie par le potentiomètre ;
   f) Sauvegarder le programme sous le nom **CAN_1.asm**.

4) **Quatrième étape : l'écran LCD**
   a) Faire le schéma électronique après avoir déterminer sur quel PORT le bus de données du LCD doit être câblé et sur quelles broches les bits RS et E doivent être câblés. **Toutes ces broches doivent être configurées en sortie**
   b) Faire le schéma de câblage ;
   c) Faire un organigramme de temporisation dont la durée doit être au minimum de 30ms.
   d) Vérifier sous MPLAB que la durée est correcte avec l'outil StopWatch;

e) Consulter la documentation technique de l'écran puis faire un organigramme incluant la procédure d'initialisation de l'écran et l'affiche de **ESME_2023** sur la première ligne ; il faut utiliser la temporisation précédente comme sous-fonction dans le programme principal ;
f) Traduire l'organigramme ci-dessus en langage assembleur puis vérifier le fonctionnement sous MPALB en utilisant l'outil Watch et en y affichant le PORT sur lequel est connecté le bus de données du LCD ;
g) Programmer le µC et vérifier que le message s'affiche correctement.
h) Sauvegarder le programme sous le nom **LCD_Ligne1** ;
i) Modifier le programme **LCD_Ligne1** pour afficher le message sur la deuxième ligne.
j) Programmer le µC puis vérifier l'affichage
k) Sauvegarder le programme sous le nom **LCD_Ligne2**.

5) Cinquième étape : Conversion Binaire/DCB

a) Etablir l'organigramme relatif à cette conversion ;
b) Ecrire le programme en langage assembleur correspondant à l'organigramme ci-dessus ;
c) Simuler le programme sous MPLAB et vérifier le bon fonctionnement. Pour ce faire, il suffit de déclarer une variable Nombre_Binaire et trois variables Centaine, Dizaine et Unite. Au début du programme, donner à Nombre_Binaire différentes valeurs entre 0 et 255, initialiser les trois autres variables à zéro et vérifier qu'à la fin du programme, les variable Centaine, Dizaine et Unite sont initialisées correctement.

6) Sixième étape : DCB/ASCII

a) Etablir l'organigramme relatif à cette conversion ; Dans ce cas, il suffit de convertir les variables Centaine, Dizaine et Unite ci-dessus en code ASCII ;
b) Ecrire le programme en langage assembleur correspondant à l'organigramme ci-dessus ;
c) Simuler le programme sous MPLAB et vérifier le bon fonctionnement en envoyant les trois variables ASCII sur le PORT sur lequel est câblé le bus de données de l'écran LCD ;
d) Programmer le µC puis vérifier l'affichage.

7) Septième étape : à venir