Aix*Marseille université

Faculté des Sciences

Informatique industrielle

Travaux dirigés

Introduction au langage assembleur

Le langage assembleur est celui qui sera utilisé en TP pour programmer les cartes PIC. Dans ce TD, on se familiarisera avec ce langage en examinant ses liens avec un langage de programmation de type texte structuré comme Matlab.

1 Objectifs

-Comparer le langage assembleur et le langage Matlab ;
-Comprendre la le lien entre le langage assembleur et la structure du PIC

2 Exercices

2.1 Exercice 1

Voici quelques instructions assembleur. Associez chaque instruction à sa signification.

1) movlw	A) déplacer le contenu du registre W dans un autre registre
2) movwf	B) déplacer le contenu d'un registre dans un autre registre
3) movff	C) déplacer un nombre dans le registre w

Julien Marot

2.2 Exercice 2

Traduisez les lignes de code en langage assembleur ci-dessous en langage Matlab :

movlw (07)h

movwf PORTA

movff PORTA, PORTB

2.3 Exercice 3

Traduisez les lignes de code assembleur ci-dessous en langage Matlab :

```
        movlw (00)h
        movwf PORTA
label   incf PORTA
        movff PORTA, PORTB
        goto label
```

2.4 Exercice 4

Traduisez les lignes de code en langage Matlab ci-dessous en langage assembleur:

```
PORTC=3;
if (PORTA(1)==0)
   PORTB=PORTC;
elseif (PORTA(1)==1)
   PORTD=PORTC;
end
```

Pour cela vous choisirez des instructions assembleur parmi les suivantes :

- movlw, movwf, movff,
- btfsc (bit test file skip if clear),
- goto
- return

Julien Marot

Aix*Marseille université

Faculté des Sciences

Mise à niveau
Informatique industrielle

Travaux Dirigés

Gestion d'une interruption externe

Julien Marot

L'objectif est d'écrire un programme qui fasse inverser l'état d'une LED branchée sur la broche 3 du PORTB, à chaque fois qu'il y a un front montant sur la broche 2 du PORTB (attention la broche 1 correspond à RB0, la broche 2 à RB1, la broche 3 à RB2, la broche 4 à RB3). Vous vous aiderez de la Figure 1 qui décrit le registre configurant les interruptions.

Configuration de l'interruption

1) Configurez le registre INTCON pour autoriser les interruptions externes.
2) Quel bit faut-il tester pour vérifier si l'interruption a eu lieu, et remettre à 0 après le programme d'interruption ?
3) S'agit-il d'une interruption synchrone ou asynchrone ?

Algorigramme et programme

4) Faites un algorigramme, avec d'un côté le programme principal, et d'un autre côté le programme d'interruption.
Le programme principal doit contenir : un début ; l'initialisation du PORTB et du registre TRISB, notamment la gestion des entrées sorties numériques, la configuration des interruptions (INTCON), une boucle, et une fin.
Le programme d'interruption doit contenir :

— un début,
— la sauvegarde du contexte,
— un test sur la valeur du flag d'interruption,
— la remise à 0 du flag d'interruption, l'inversion de la LED,
— la restauration du contexte et le retour au programme principal.

5) Complétez le code suivant :
un _ _ correspond à une opérande, trouvez l'opérande qui convient.

```
; Interruption externe
LIST P=18F4520
#include <P18F4520.inc>
#include <CONFIG.inc>

;----- Déclaration de variables
CBLOCK 0x00
W_TEMP  : 1 ;Résevations de 3 octects dans la RAM
STATUS_TEMP : 1 ;Pour la sauvegarde du contexte
BSR_TEMP : 1
ENDC
;----- Programme
org h'0000'                         ;Adresse de début du programme sur Reset
goto init

org h'0008'                         ;Adresse de début du programme sur Reset
goto routine_int

init
    clrf  PORTB                 ;Remise à zéro des bascules D du port B
movlw _ _
movwf _ _                           ;le port B est défini en sortie, sauf RB1
movlw _ _
movwf _ _                           ;broches 1 à 4 du PORTB en E/S numériques
movlw _ _
movwf _ _                          ;Autorisation Générale des IT et INT0 IT

boucle _ _                          ;Ne rien faire
goto boucle                        ;Boucle infini

;----- Routine d'interruption
routine_int
movwf  W_TEMP                       ;Sauvegarde de W
movff  STATUS, STATUS_TEMP           ;Sauvegarde de STATUS
movff  BSR, BSR_TEMP               ; Sauvegarde de BSR
btfsc  INTCON,1                   ;INT0IF == 1 _ _
rcall  _ _                         ;traitement de l'interruption INT0
movff BSR_TEMP, BSR               ;Restauration de BSR
movff  W_TEMP, WREG               ;Restauration de W
movff STATUS_TEMP, STATUS              ;Restauration de STATUS
retfie                       ;Retour au programme principal

;----- Interruption par evenement extérieur
interruption_INT0
bcf  _ _                            ;Suppression du flag d'interruption
movlw  _ _
xorwf  _ _                         ;Inversion de l'état de la broche RB2 du PORTB
return
END
```

REGISTER 9-1: INTCON REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
bit 7							bit 0

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low priority peripheral interrupts
0 = Disables all low priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Note: A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

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 1 – Registre INTCON

Aix*Marseille université

Faculté des Sciences

Mise à niveau
Informatique industrielle

Travaux Dirigés

Modulation de largeur d'impulsion :
PWM pulse width modulation

Julien Marot

1 Généralités

Objectifs :

On cherche à utiliser le "module CCP" du microcontrôleur pour générer un signal rectangulaire sur la broche RC2. La période du PWM est fixée à une fréquence de 600 Hz et le rapport cyclique à 0.5.
Des figures en annexe vous aideront à répondre aux questions.

Questions :
- Expliquez comment les modules TIMER2 et CCP1 fonctionnent ensemble pour produire un signal rectangulaire de période et de rapport cyclique donné ;
- Identifiez les registres à initialiser et les valeurs associées ;
- Construisez l'algorigramme et écrivez le programme assembleur.

2 Question 1 : les modules TIMER2 et CCP1

TABLE 15-5: REGISTERS ASSOCIATED WITH PWM AND TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset Values on page
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	49
RCON	IPEN	SBOREN	—	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$	48
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	52
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	52
IPR1	PSPIP(1)	ADIP	RCIP	TXIP	SSPIP	CCP1IP	TMR2IP	TMR1IP	52
TRISB	PORTB Data Direction Control Register								52
TRISC	PORTC Data Direction Control Register								52
TMR2	Timer2 Register								50
PR2	Timer2 Period Register								50
T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0	50
CCPR1L	Capture/Compare/PWM Register 1, Low Byte								51
CCPR1H	Capture/Compare/PWM Register 1, High Byte								51
CCP1CON	P1M1(1)	P1M0(1)	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0	51
CCPR2L	Capture/Compare/PWM Register 2, Low Byte								51
CCPR2H	Capture/Compare/PWM Register 2, High Byte								51
CCP2CON	—	—	DC2B1	DC2B0	CCP2M3	CCP2M2	CCP2M1	CCP2M0	51
ECCP1AS	ECCPASE	ECCPAS2	ECCPAS1	ECCPAS0	PSSAC1	PSSAC0	PSSBD1(1)	PSSBD0(1)	51
PWM1CON	PRSEN	PDC6(1)	PDC5(1)	PDC4(1)	PDC3(1)	PDC2(1)	PDC1(1)	PDC0(1)	51

Legend: — = unimplemented, read as '0'. Shaded cells are not used by PWM or Timer2.
Note 1: These bits are unimplemented on 28-pin devices; always maintain these bits clear.

FIGURE 1 – Tous les registres associés au PWM (p. 146 datasheet)

Le registre TMR2 est un registre qui est incrémenté, le registre PR2 configure la période du TIMER2 et indirectement de la PWM, le registre T2CON configure le TIMER 2.
Le registre CCP1CON configure le module CCP, CCPR1L en association avec CCP1CON fixe le rapport cyclique du signal rectangulaire.

2.1 les registres TMR2, PR2, T2CON

TMR2 est un registre de 8 bits qui est incrémenté, comme le registre TMR0 pour le TIMER0. PR2 est un registre de 8 bits qui configure la période du TIMER2 et de la PWM par la formule suivante :

```
PWM Period = (PR2+1)x 4 x TOSC x (TMR2 Prescale Value)
```

où `TOSC` est la période de l'horloge, 0.25 10^{-6} sec. et où `TMR2 Prescale Value` est déterminée par le registre T2CON (voir Fig. 2).

Comment faire pour :

— mettre TIMER2 sur on, choisir un postscaler de 1 et un prescaler de 16 ?

— choisir une fréquence de 600 Hz ?

2.2 les registres CCP1CON et CCPR1L

Le registre CCP1CON configure le module CCP. En particulier les bits 5 et 4 sont associés aux 8 bits de CCPR1L pour fixer le rapport cyclique de la PWM.

Le registre CCPR1L permet de déterminer, avec le CCP1CON, le rapport cyclique de la PWM. Pour cela on applique la formule suivante :

```
PWM Duty Cycle = (CCPR1L:CCP1CON<5:4>) x TOSC x (TMR2 Prescale Value)
```

où `PWM Duty Cycle` est la durée de l'état haut de la PWM, exprimée en sec. D'après cette définition, le rapport cyclique (noté α) de la PWM est donné par :

$$\alpha =$$

```
(PWM Duty Cycle) / (PWM Period)
```

D'après la définition des bits du registre CCP1CON (voir la description Fig. 3), et supposant que :

— par défaut tous les bits du CCP1CON sont à 0,

— TMR2 Prescaler value vaut 16,

que faut-il faire pour :

— placer le module CCP1 en mode PWM ?

— fixer le rapport cyclique de la PWM à 0.5 ?

3 Question 2 : listez les registres à initialiser et les valeurs associées

4 Question 3 : construisez l'algorigramme et le programme en assembleur

4.1 Algorigramme

L'algorigramme comprend essentiellement une phase d'initialisation, et une boucle d'attente.

4.2 Programme en assembleur

Complétez le code suivant :
un _ _ correspond à une opérande, trouvez l'opérande qui convient.

```
    LIST P=18F4520
    #include <P18F4520.inc>

; Initialisation du vecteur RESET
    org h'0000'
    goto init

; Initialisation du microcontroleur
;
; config PORTC

init
    clrf PORTC
    movlw _ _
    movwf TRISC ; RC0 à RC7 en Sortie

; Configuration du module PWM
    movlw _ _
    movwf PR2 ; frequence 600Hz

    movlw _ _
    movwf CCPR1L ; Rapport cyclique .5

    movlw _ _
    movwf CCP1CON ; Rapport cyclique .5, mode PWM tout en active low, single output


; Configuration du Timer2
    movlw _ _ ;
    movwf T2CON ; TMR2 On, post=1, pres=16

; programme principal ;
    main nop
    goto main

    END
```

Figures

REGISTER 13-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **T2OUTPS3:T2OUTPS0**: Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON**: Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS1:T2CKPS0**: Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

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 2 – Le registre T2CON : configuration du TIMER 2 (p. 133 datasheet)

REGISTER 16-1: CCP1CON REGISTER (ECCP1 MODULE, 40/44-PIN DEVICES)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
P1M1	P1M0	DC1B1	DC1B0	CCP1M3	CCP1M2	CCP1M1	CCP1M0
bit 7							bit 0

bit 7-6 **P1M1:P1M0:** Enhanced PWM Output Configuration bits

If CCP1M3:CCP1M2 = 00, 01, 10:

xx = P1A assigned as Capture/Compare input/output; P1B, P1C, P1D assigned as port pins

If CCP1M3:CCP1M2 = 11:

00 = Single output: P1A modulated; P1B, P1C, P1D assigned as port pins
01 = Full-bridge output forward: P1D modulated; P1A active; P1B, P1C inactive
10 = Half-bridge output: P1A, P1B modulated with dead-band control; P1C, P1D assigned as port pins
11 = Full-bridge output reverse: P1B modulated; P1C active; P1A, P1D inactive

bit 5-4 **DC1B1:DC1B0**: PWM Duty Cycle bit 1 and bit 0

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPR1L.

bit 3-0 **CCP1M3:CCP1M0**: Enhanced CCP Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCP module)
0001 = Reserved
0010 = Compare mode, toggle output on match
0011 = Capture mode
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, initialize CCP1 pin low, set output on compare match (set CCP1IF)
1001 = Compare mode, initialize CCP1 pin high, clear output on compare match (set CCP1IF)
1010 = Compare mode, generate software interrupt only, CCP1 pin reverts to I/O state
1011 = Compare mode, trigger special event (ECCP resets TMR1 or TMR3, sets CC1IF bit)
1100 = PWM mode; P1A, P1C active-high; P1B, P1D active-high
1101 = PWM mode; P1A, P1C active-high; P1B, P1D active-low
1110 = PWM mode; P1A, P1C active-low; P1B, P1D active-high
1111 = PWM mode; P1A, P1C active-low; P1B, P1D active-low

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 3 – Le registre CCP1CON : configuration du module CCP (p. 147 datasheet)