POLYTECH MARSEILLE
Aix*Marseille Université

GROUPE : ADIRA ALIYYA
HASSANEIN ALZAHRA

CLIENT : LE VAN SUU AUGUSTE

Rapport Final

PROJET D'ARCHITECTURE DES SYSTÈMES : ROBOT TÉLÉCOMMANDÉ PAR COMMANDE VOCALE

Auteurs: ADIRA Aliyya et HASSANEIN Alzahra
Client : LE VAN SUU Auguste
Copies : TOUCHARD François
Version : 1.0

Signatures de l'équipe de production:

Signature du client:

SOMMAIRE :

1. INTRODUCTION

OBJET :

Notre projet d'architecture des ordinateurs consiste à réaliser un système robot télécommandé avec des communications hautes-fréquences en intégrant une reconnaissance vocale. Nous avons environ deux mois pour développer le produit qui répond aux exigences de ce cahier des charges, et le livrer au client.

CONTEXTE :

Le projet que nous réalisons s'inscrit dans le cadre du cours "Architecture des ordinateurs" de la troisième année du cycle d'ingénieur, de la filière "Informatique, Réseaux, Multimédia" à POLYTECH Marseille. Ce cours est dispensé par M. François TOUCHARD.

Notre tuteur et client est M. Auguste LE VAN SUU. Nous sommes deux binômes d'étudiants en charges de ce projet. Ce cahier des charges est réalisé par le binôme composé de ADIRA Aliyya et HASSANEIN Alzahra.

L'objectif est avant tout d'avoir une première approche de la gestion de projet d'une manière professionnelle grâce à la relation fournisseur-client que nous devons entreprendre tout au long du projet. Il faudra respecter les exigences demandées, fournir les différents livrables tels que le cahier des charges, le cahier de conception, l'analyse fonctionnelle, le manuel d'utilisation ou encore la garantie. Il faudra également effectuer un suivi client ainsi qu'effectuer une présentation et démonstration lors du rendu du rapport final.

Ce projet va nous permettre de nous familiariser avec l'utilisation de matériel, d'appliquer nos connaissances acquises tout au long de l'année et d'améliorer la gestion d'un projet en équipe.

Le robot à développer dans ce projet sera capable de se déplacer dans les couloirs de l'école Polytech Marseille grâce à des mouvements simples. La commande sera possible par commande vocale et par le clavier d'un ordinateur.

NB : Les détails de l'analyse fonctionnelle, du cahier des charges et du cahier de conception sont dans des fichiers distincts, de ce fait nous n'avons pas réécrit l'intégralité des informations ici.

PLANNING INITIAL :

MARS 2017

Semaine	5			9					10						11							12							13				
	V	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J		
	3/2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30		
Projet	Présenté			Cahier des charges, Analyse F. & Planning									Prise en main des logiciels et composants																				
Aliyya A.				CdC									Reconnaissance Vocale																				
Alzahra H.				A.F et Planning									PIC16F84 et émission																				
Rendez-vous				Présentation						Rendu !															Séance								

AVRIL 2017

Semaine	13			14							15							16								17						
	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D
	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Projet	Programmation																	Correction des problèmes								Jeux de tests						
Aliyya A.	Reconnaissance Vocale																	Reconnaissance Vocale								RV et Robot						
Alzahra H.	PIC16F84 et émission																	PIC16F84 et émission								Oscilloscope						
Rendez-vous							Séance																		Séance								

MAI 2017

Semaine	18							19							20							21							22		
	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Projet	Mise en commun du travail avec la partie réception														Derniers ajustements et tests											Rendu			Soutenance		
Aliyya A.	RV et Robot														Assemblage du tout											Rendu			Soutenance		
Alzahra H.	Oscilloscope et tests emetteurs/recepteurs														Test en HF																	
Rendez-vous									Séance						Séance																	

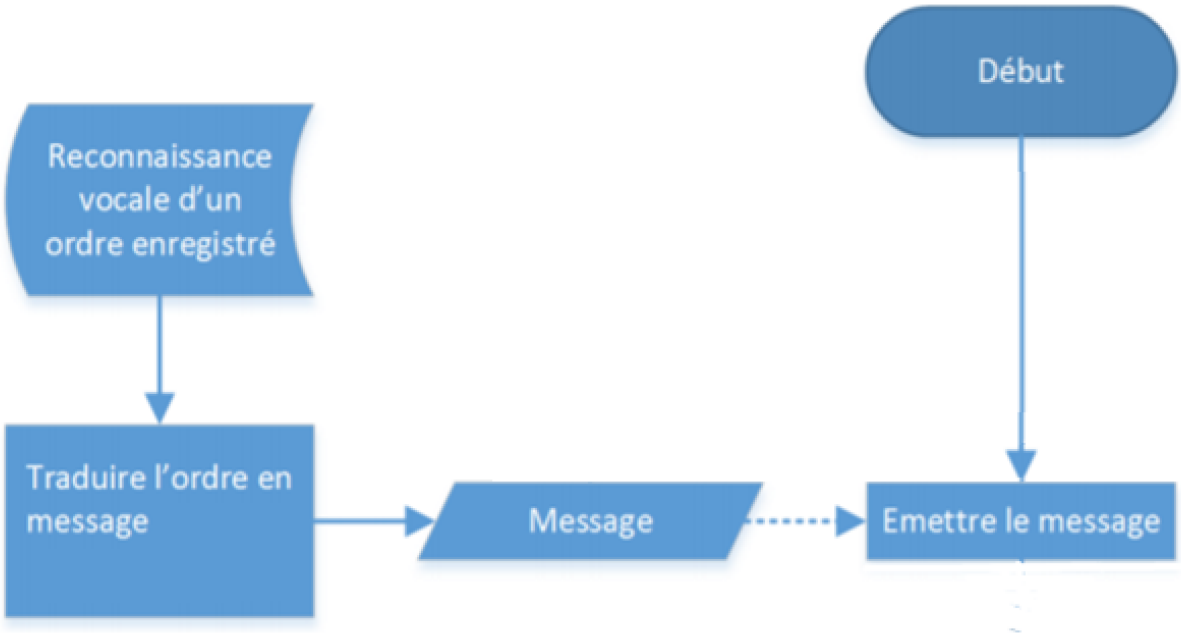
Jour Férié
Week-end
Dates Importantes

2. CONCEPTION

ANALYSE :

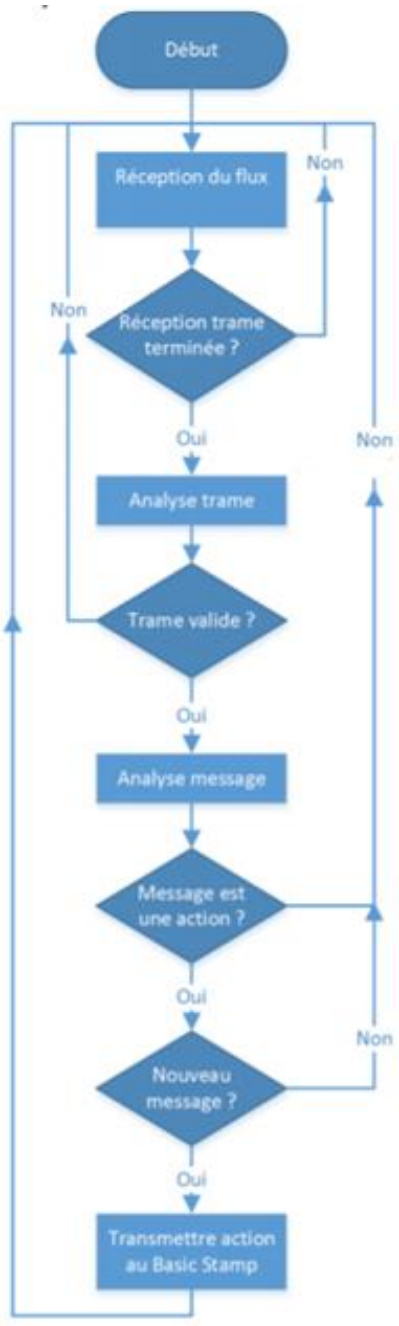
Pour effectuer l'émission et la réception, nous avons établi un système qui se retrouvent dans ces deux algorithmes. L'émission et la réception se font sur des PIC16F84 positionnés sur leur carte respective d'émission et de réception, en respectant le montage (en annexe).

Algorithme Emission:



Ici dans notre cas, l'émission est distinguée par le nombre de pulse (front montant) définit pour chaque ordre (le détail de cela est défini dans le cahier de conception).
Chaque message est aussi espacé de 1 seconde, pour pouvoir les distinguer.

Algorithme Réception:



Ici, nous avons donc 2 boucles imbriqués la première attendant le premier flag, la seconde enclenchant la lecture du message en comptant le nombre de pulse reçut grâce à l'interruption RB0 prévu dans le code d'assemblage du pic.

Pour rappel :

En émission nos entrées de la RV se font sur les broches RA0, RA1, RA2 (ce sont les mêmes que les broches de sorties de la réception) et notre sortie se fait sur RB7 qui dessert l'entrée de l'émetteur TX433-SAW mais également un circuit à led pour test.

En réception l'entrée est sur RB0 et se fait via le récepteur RF 290 A-5 S.

PLANNIFICATION :

Contrairement à la planification initiale qui était :

- Reconnaissance Vocale (Aliyya Adira)
- Émission (Alzahra Hassanein)
- Réception (Dylan Dia)
- Programmation du Robot (Mama Dembele)

Le binôme Adira/Hassanein fut chargé de la partie réception en fin de projet, afin de pouvoir finir au mieux la majeure partie du projet.

PLANNING FINAL :

MARS 2017

Semaine	5			9				10						11							12							13					
	V	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J
	3/2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Projet	Présenté			Cahier des charges, Analyse F. & Planning									Prise en main des logiciels et composants																			
Aliyya A.				CdC									Reconnaissance Vocale																			
Alzahra H.				A.F et Planning									PIC16F84 et émission																			
Rendez-vous				Présentation						Rendu !														Séance							

AVRIL 2017

Semaine	13			14						15						16							17						
	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D
	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Projet	Prise en main des logiciels et composants									Programmation															Jeux de tests						
Aliyya A.	Reconnaissance Vocale															PIC16F84 et émission															
Alzahra H.	PIC16F84 et émission																														
Rendez-vous							Séance																	Séance							

MAI 2017

Semaine	18							19							20							21							22		
	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S	D	L	M	M
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Projet	Jeux de tests									Mise en commun avec la partie réception & reprogrammation												Reprogrammation, Tests, Assemblage									
Aliyya A.	RV et Robot														Oscilloscope et tests recepteurs														Test et Assemblage		Soutenance
Alzahra H.	Oscilloscope et tests emetteurs & trames														Oscilloscope et tests recepteurs																
Rendez-vous									Séance						Séance																

Jour Férié
Week-end
Dates Importantes

3. CONCLUSION

En bilan :

- Pour la reconnaissance vocale, les commandes reçus par la voix sont bien analysés et les bits envoyés sont conformes aux demandes
- Pour l'émission, on observe bien avec notre test à l'oscilloscope une cohérence entre l'analyse des bits reçus et le nombre de pulse (le signal) envoyés à la réception.
- Pour la réception, le même test qu'à l'émission confirme la bonne réception du signal, et un jeux de test de led en sortie confirme le bon décodage du signal et l'envoi des trames de bits demandés et nécessaires pour le robot, cependant en HF le signal reçu étant brouillé par des parasites, la commande s'execute avec difficulté.
- Pour la programmation des mouvements du robot, l'éxecution des commandes se fait de manière claire et a été vérifié grâce au branchement de la reconnaissance vocale directement à celui-ci.

En définitive, il nous manquerait 1 voire 2 semaine afin de pouvoir finir convenablement le robot et effacer la présence des parasites en effectuant des tests avec d'autres émetteurs et d'autres récepteurs, ou en modifiant le tempo d'émission ou la fréquence de sortie des bits en réception.

Pour plus d'information sur la partie commande du robot, et réception, vous pourrez également consulter le rapport de DEMBELE Mama et DIA Dylan (notre binôme sur ce projet).

4. ANNEXES

REFERENCES :

Nous avons utilisé pour ce projet les sites :

- https://www.abcelectronique.com/bigonoff/index.php
- http://fabrice.sincere.pagesperso-orange.fr/cm_electronique/pic_accueil.htm

ainsi que l'ensemble de la documentation fournit sur le matériel.

CODES SOURCES :

Basic Stamp :

```
' {$STAMP BS2}
' {$PBASIC 2.5}


HIGH 12
HIGH 13

'Definition des constantes
left_wheel CON 12
right_wheel CON 13


vitesse_left CON 1000
vitesse_right CON 100

vitesse_recule_left  CON  100
vitesse_recule_right  CON 400


 'PLUS DE MODIF
 'P7 -> GP16
 'P6 -> GP15
 'P5 -> GP14
'Definition des VARIABLES :



command VAR Word

cpt VAR Word

'La fonction principale

main :

bit_1  VAR  IN4
INPUT 4
bit_2  VAR  IN5
INPUT 5
bit_3  VAR  IN6
INPUT 6
bit_4  VAR  IN7
INPUT 7
 command = 0
```

```
  IF bit_4 = 1 THEN add_bit_4
add_bit_4_return :

  IF bit_3 = 1 THEN add_bit_3
add_bit_3_return :

  IF bit_2 = 1 THEN add_bit_2
add_bit_2_return :

  bit_1 = 0

  IF bit_2<>0 | bit_3<>0 | bit_4<>0 THEN
    DEBUG DEC command, CR
  ENDIF

  IF command = 1   THEN forward
  IF command = 2   THEN backward
  IF command = 3   THEN left
  IF command = 4   THEN right
  IF command = 5   THEN stoppe

  GOTO main


'Les fonctions pour definir les
'mouvements

forward :
FOR cpt = 0 TO 30
    PULSOUT left_wheel, vitesse_left
    PULSOUT right_wheel, vitesse_right
NEXT
GOTO main


backward :
  FOR cpt = 0 TO 30
    PULSOUT left_wheel, vitesse_recule_left
    PULSOUT right_wheel, vitesse_recule_right
  NEXT
  GOTO main

right :
  FOR cpt = 0 TO 30
    PULSOUT left_wheel, 0
    PULSOUT right_wheel, vitesse_right
  NEXT
  GOTO main

left :
  FOR cpt = 0 TO 30
    PULSOUT left_wheel, vitesse_left
    PULSOUT right_wheel, 0
  NEXT
  GOTO main
```

```
stoppe :
  PULSOUT left_wheel, 0
  PULSOUT right_wheel, 0
  GOTO main

'Fonctions permettant de mettre à jour
'la commande.

add_bit_4 :
  command = command + 1
  GOTO add_bit_4_return

add_bit_3 :
  command = command + 2
  GOTO add_bit_3_return

add_bit_2 :
  command = command + 4
  GOTO add_bit_2_return
```

Assembleur Émission :

```
;   Ce fichier est la base de départ pour une programmation avec
;   le PIC 16F84. Il contient les informations de base pour
;   démarrer.
;
;   Si les interruptions ne sont pas utilisées, supprimez les lignes
;   entre ORG 0x004 et l'étiquette init. De plus, les variables
;   w_temp et status_temp peuvent être supprimées.
;
;    Fichier requis: P16F84.inc
;    NOM: HASSANEIN, ADIRA, DEMBELE,DIA
;    Date:  25/05/2017
;    Version:   5


    LIST      p=16F84           ; Définition de processeur
    #include <p16F84.inc>        ; Définitions des constantes
    radix dec                   ; on travaille en décimal par défaut

    __CONFIG   _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

;                       ASSIGNATIONS

 OPTIONVAL    EQU    H'40'          ; Valeur registre option
                        ; Résistance pull-up ON
                        ; Interrupt flanc montant RB0
                        ; Préscaler timer à 2 (exemple)

 INTERMASK    EQU    H'90'        ; Masque d'interruption
                        ; Interruptions sur RB0 (exemple)
```

```
;                   DECLARATIONS DE VARIABLES

   CBLOCK 0x00C              ; début de la zone variables
   w_temp :1                 ; Zone de 1 byte
   status_temp : 1           ; zone de 1 byte
   cmpt1                     ; compteur de boucles 1
   cmpt2                     ; compteur de boucles 2
   cmpt3                     ; compteur de boucles 3
   cmpt4                     ; compteur pour nb message
  nbfront
   ENDC                      ; Fin de la zone

;                  DEMARRAGE SUR RESET

   org 0x000                 ; Adresse de départ après reset
     goto    init            ; Adresse 0: initialiser

;                  INITIALISATIONS

init
   bcf       STATUS,RP0      ; sélectionner banque 0
   clrf          PORTA        ; Sorties portA à 0
   clrf          PORTB        ; sorties portB à 0
   bsf       STATUS,RP0      ; sélectionner banque 1
   bsf       TRISA,0         ;entrée
   bsf       TRISA,1         ;entrée
   bsf       TRISA,2         ;entrée
   bcf       TRISB,7         ;sortie
   bcf       STATUS,RP0      ; sélectionner banque 0
   goto start

;                  SOUS-ROUTINE DE TEMPORISATION

tempo                                ; retard de 500 µs. = 0.5ms = 166
         movlw 166
         movwf cmpt4
boucle4
         nop
         decfsz  cmpt4,f
         goto boucle4
         return

tempo2                                    ;0.5s
         movlw  2                         ; pour 2 boucles
         movwf  cmpt3                     ; initialiser compteur3
boucle3
         clrf      cmpt2                  ; effacer compteur2
boucle2
         clrf      cmpt1                  ; effacer compteur1
```

```
boucle1
        nop                                             ; perdre 1 cycle
        decfsz cmpt1,f                          ; décrémenter compteur1
        goto    boucle1                         ; si pas 0, boucler
        decfsz cmpt2,f                  ; si 0, décrémenter compteur 2
        goto    boucle2                         ; si cmpt2 pas 0, recommencer boucle1
        decfsz cmpt3,f                          ; si 0, décrémenter compteur 3
        goto    boucle3                         ; si cmpt3 pas 0, recommencer boucle2
        return                                  ; retour de la sous-routine


;               PROGRAMME PRINCIPAL

start
        btfss PORTA, 0          ;RA0 = 1
        goto reculer_droite     ;RA0 = 0
        goto avancer_gauche_stop


reculer_droite                  ; RA0 = 0
        btfss PORTA, 1          ;  RA1 = 1
        goto droite             ;RA1 = 0
        goto reculer


avancer_gauche_stop             ;RA0=1
        btfss PORTA, 1
        goto avancer_stop       ;RA1 = 0
        goto gauche             ;RA1 = 1


avancer_stop                    ; RA0 = 1 RA1 = 0
        ;call tempo
        btfss PORTA, 2          ;RA2 = 1
        goto avancer_front      ;RA2 = 0
        goto stop_front

droite                          ;RA0 = 0 RA1 = 0
        btfss PORTA, 2          ; RA2 = 1
        goto start
        goto droite_front

reculer                         ;RA0 = 0 RA1 = 1
        btfss PORTA, 2          ; RA2 = 1
        goto reculer_front
        goto start

gauche                          ;RA0 = 1 RA1 = 1
        btfss PORTA, 2          ;RA2 = 1
        goto gauche_front       ;RA2=0
        goto start
```

```
gauche_front
        ;On envoie 16 fronts (16 bits)
        ;RB7 a 8 bits à 1
        movlw 8
        movwf nbfront
envoie_front4
        bsf PORTB, 7   ; RB7 = 1
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        bcf PORTB, 7   ; RB7 = 0
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        decfsz nbfront, f
        goto envoie_front4
        goto temporisation

avancer_front
        ;On envoie 8 fronts (8 bits)
        ;RB7 a 4 bits à 1
        movlw 4
        movwf nbfront
envoie_front1
        bsf PORTB, 7   ; RB7 = 1
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        bcf PORTB, 7   ; RB7 = 0
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        decfsz nbfront, f
        goto envoie_front1
        goto temporization

reculer_front
        ;On envoie 12 fronts (12 bits)
        ;RB7 a 6 bits à 1
        movlw 6
        movwf nbfront
envoie_front2
        bsf PORTB, 7   ; RB7 = 1
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        bcf PORTB, 7   ; RB7 = 0
        call tempo
        call tempo
```

```
        call tempo
        call tempo      ;2ms
        decfsz nbfront, f
        goto envoie_front2
        goto temporisation

droite_front
        ;On envoie 20 bits
        ;RB7 a 10 bits à 1
        movlw 10
        movwf nbfront
envoie_front3
        bsf PORTB, 7   ; RB7 = 1
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        bcf PORTB, 7   ; RB7 = 0
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        decfsz nbfront, f
        goto envoie_front3
        goto temporisation

stop_front
        ;On envoie 30 fronts
        ;RB7 a 15 bits à 1
        movlw 15
        movwf nbfront
envoie_front5
        bsf PORTB, 7   ; RB7 = 1
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        bcf PORTB, 7   ; RB7 = 0
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms
        decfsz nbfront, f
        goto envoie_front5
        goto temporisation

temporisation
        call tempo2
        ;call tempo2
        ;call tempo2
        call tempo2 ; 2secondes
        goto start

    END             ; directive fin de programme
```

Assembleur Réception :

```
;   Ce fichier est la base de départ pour une programmation avec
;   le PIC 16F84. Il contient les informations de base pour
;   démarrer.
;
;   Si les interruptions ne sont pas utilisées, supprimez les lignes
;   entre ORG 0x004 et l'étiquette init. De plus, les variables
;   w_temp et status_temp peuvent être supprimées.
;
;    Fichier requis: P16F84.inc
;    NOM: HASSANEIN, ADIRA, DEMBELE,DIA
;    Date:  25/05/2017
;    Version:   7


    LIST      p=16F84           ; Définition de processeur
    #include <p16F84.inc>        ; Définitions des constantes
    radix dec                   ; on travaille en décimal par défaut

    __CONFIG   _CP_OFF & _WDT_OFF & _PWRTE_OFF & _XT_OSC


;                       ASSIGNATIONS

OPTIONVAL    EQU    H'40'          ; Valeur registre option
                        ; Résistance pull-up ON
                        ; Interrupt flanc montant RB0
                        ; Préscaler timer à 2 (exemple)

INTERMASK    EQU    H'90'        ; Masque d'interruption
                        ; Interruptions sur RB0 (exemple)


;                   DECLARATIONS DE VARIABLES

CBLOCK 0x00C                ; début de la zone variables
    w_temp :1                ; Zone de 1 byte
    status_temp : 1             ; zone de 1 byte
    cmpt1                   ; compteur de boucles 1
    nbfront
    bouclea
    ENDC                     ; Fin de la zone
```

```
;                   DEMARRAGE SUR RESET

    org 0x000                   ; Adresse de départ après reset
      goto    init              ; Adresse 0: initialiser



;                   ROUTINE INTERRUPTION


   ;sauvegarder registres
   ;---------------------

   ORG 0x004                    ; adresse d'interruption
   movwf        w_temp          ; sauver registre W
   swapf        STATUS,w        ; swap status avec résultat dans w
   movwf        status_temp     ; sauver status swappé

   incf         nbfront,1
   ;bsf         PORTB,2
   bcf    INTCON,INTF           ; effacer flag interupt RB0

      ;restaurer registres

   swapf        status_temp,w ; swap ancien status, résultat dans w
   movwf        STATUS          ; restaurer status
   swapf        w_temp,f        ; Inversion L et H de l'ancien W
                                ; sans modifier Z
   swapf        w_temp,w        ; Ré-inversion de L et H dans W
                                ; W restauré sans modifier status
   retfie                       ; return from interrupt


;                   INITIALISATIONS

init
    bcf     STATUS,RP0      ; sélectionner banque 0
    clrf    PORTA           ; Sorties portA à 0
    clrf    PORTB           ; sorties portB à 0

    bsf     STATUS,RP0          ; sélectionner banque 1
    bcf     TRISA,0             ;sortie
    bcf     TRISA,1             ;sortie
    bcf     TRISA,2             ;sortie
    bsf     TRISB,0             ;entrée


    bcf     STATUS,RP0          ; sélectionner banque 0
    clrf    INTCON              ; tout a zerro (interuptions)
    bsf     INTCON,INTE         ; interruption sur RB0 (4)
    clrf    nbfront             ; Le message est vide
    goto start
```

```
;                SOUS-ROUTINE DE TEMPORISATION

tempo
   movlw       250          ; pour 249*2 + 250
   movwf       cmpt1        ; initialiser compteur1
boucle1
   nop
   decfsz      cmpt1 , f    ; décrémenter compteur1
   goto        boucle1        ; si pas 0, boucler
   return            ; retour de la sous-routine


;                PROGRAMME PRINCIPAL

start
        clrf    PORTA
        bsf     INTCON,GIE ;activation interruption
        ;call tempo
        movlw 0
        movwf nbfront

attente                                 ;boucle d'attente du premier flag
        btfsc   nbfront,0
        goto    attente2
        goto    attente

attente2                                ;boucle d'attente du message
        call tempo
        call tempo
        call tempo
        call tempo  ;2ms    REPETER 21fois pour avoir 48ms de delay

        goto    traitement

traitement
        bcf     INTCON,GIE
        movlw 3
        subwf   nbfront,1 ;nbfront - W
        decfsz nbfront, f; nbfront - 4
        goto    reculer
        goto    avancer2

avancer2
        bsf PORTA, 0
        nop
        nop
        bcf PORTA, 1
        nop
        nop
        bcf PORTA, 2
        nop
        nop
        call tempo
        call tempo
        goto start
```

```
reculer
        movlw  1
        subwf  nbfront,1
        decfsz nbfront,f ;nbfront - 6
        goto   gauche
        goto   reculer2


reculer2
        bcf PORTA, 0
        nop
        nop
        bsf PORTA, 1
        nop
        nop
        bcf PORTA, 2
        nop
        nop
        call tempo
        call tempo
        goto start


gauche
        movlw  1
        subwf  nbfront,1
        decfsz nbfront,f ;nbfront - 8
        goto   droite
        goto   gauche2


gauche2
        bsf PORTA, 0
        nop
        nop
        bsf PORTA, 1
        nop
        nop
        bcf PORTA, 2
        nop
        nop
        call tempo
        call tempo
        goto start


droite
        movlw  1
        subwf  nbfront,1
        decfsz nbfront,f ;nbfront -10
        goto   stop
        goto   droite2
```

```
droite2
        bcf PORTA, 0
        nop
        nop
        bcf PORTA, 1
        nop
        nop
        bsf PORTA, 2
        nop
        nop
        call tempo
        call tempo
        goto start

stop
        movlw  5
        subwf  nbfront,1
        ;movlw 1
        ;subwf  nbfront,1
        ;decfsz         nbfront,f ;nbfront - 15
        btfsc   nbfront,0
        goto    start
        goto    stop2

stop2
        bsf PORTA, 0
        nop
        nop
        bcf PORTA, 1
        nop
        nop
        bsf PORTA, 2
        nop
        nop
        call tempo
        call tempo
        goto start
        END             ; directive fin de programme
```