Modo zampe: Esapodo (I)

```
c:\pathfi~1\cdelro~1\zampe\esap1.asm
  1         list p=16F876
  2
  3         include "P16F876.inc"
  4
  5 CONTATORE       EQU     0x20
  6 TEMP            EQU     0x30
  7 v_STATUS        EQU     0x25
  8
  9 ;RB2 e RB3 -> ZAMPA1 (JP4) (sinistra) con Sensore RC0 (JP12)
 10 ;RB4 e RB5 -> ZAMPA2 (JP2) (destra) con Sensore RC1 (JP17)
 11 ;RB0 e RB1 -> ZAMPA3 (JP3) (Centrale) con Sensore RC2 (JP18)
 12 ;Quando i sensori rilevano nero inviano un '1'. Se rilevano bianco inviano uno '0'
 13 ; 1 -> NERO
 14 ; 0 -> BIANCO
 15 ;RA4: Finecorsa sinistro (JP7)
 16 ;RA2: Finecorsa destro (JP8)
 17
 18 ;STATI IN CUI PASSA IL ROBOT NEL SUO MOVIMENTO
 19 ;STATUS 1
 20 ;       OFF_ZAMPA1
 21 ;       OFF_ZAMPA2
 22 ;       DX_ZAMPA3
 23 ; Sensore RC2 su bianco
 24
 25 ;STATUS 2
 26 ;       OFF_ZAMPA1
 27 ;       OFF_ZAMPA2
 28 ;       DX_ZAMPA3
 29 ; Sensore RC2 su nero
```

Inizieremo l'analisi dei tre programmi per il controllo di Pathfinder in modo esapodo, che si trovano sotto la directory Zampe del secondo CD-ROM. Per provare questi esercizi Pathfinder deve essere configurato con le zampe montate in modo esapodo. Dobbiamo collegare il sensore ottico con funzione di encoder della ruota destra sul connettore JP17 della scheda di controllo, il sensore ottico con funzione di encoder della ruota sinistra sul connettore JP12 e il sensore ottico di controllo delle sei zampe centrali con il connettore JP18. Per realizzare questi collegamenti dobbiamo scollegare i sensori ottici anteriori del robot che si utilizzano solamente nella configurazione con le ruote.

```
c:\pathfi~1\cdelro~1\zampe\esap1.asm
 91 ;MACRO UTILIZZATE NEL PROGRAMMA
 92 ;^^^^^^^^^^^^^^
 93 SX_ZAMPA3:      macro
 94                 bcf     PORTB, 0
 95                 bsf     PORTB, 1
 96                 ENDM
 97 DX_ZAMPA3:      macro
 98                 bsf     PORTB, 0
 99                 bcf     PORTB, 1
100                 ENDM
101 OFF_ZAMPA3:     macro
102                 bcf     PORTB, 0
103                 bcf     PORTB, 1
104                 ENDM
105 ;^^^^^^^^^^^^
106 AV_ZAMPA2:      macro
107                 bsf     PORTB, 5
108                 bcf     PORTB, 4
109                 ENDM
110 DX_ZAMPA2:      macro
111                 bcf     PORTB, 5
112                 bsf     PORTB, 4
113                 ENDM
114 OFF_ZAMPA2:     macro
115                 bcf     PORTB, 5
116                 bcf     PORTB, 4
117                 ENDM
118 ;^^^^^^^^^^^^
119 AV_ZAMPA1:      macro
```

Il primo programma che analizzeremo sarà esap1.asm. Questo programma permetterà al robot di camminare in avanti in modo esapodo. Per facilitare la lettura del programma, all'inizio dello stesso sono state generate nuove istruzioni (mediante macro) che serviranno per fermare ognuna delle zampe o attivarle in un senso o nell'altro. Osservando il robot dalla parte posteriore, chiameremo ZAMPA1 la zampa sinistra, ZAMPA2 quella destra e ZAMPA3 quella centrale. Le zampe sinistra e destra possono realizzare movimenti di avanzamento o retromarcia, e la zampa centrale è quella che si muove verso i lati per permettere l'avanzamento e la rotazione del robot.

```
c:\pathfi~1\cdelro~1\zampe\esap1.asm
135 ;*****************************************************************
136 ;Inizio del programma principale
137 INIZIO:
138         bcf     STATUS, RP1
139         bsf     STATUS, RP0
140         movlw b'00000111'
141         movwf ADCON0
142         movlw   b'00001111'
143         movwf   PORTA
144         clrf    PORTB
145         movlw   b'00000111'
146         movwf   PORTC
147         bcf     STATUS, RP0
148         clrf    PORTB
149         movlw   b'00100000'
150         movwf   PORTC
151
152 ;Routine di inizializzazione che si fa prima di iniziare il movimento. Tutti
153 ;i sensori stanno vedendo il settore nero, cioè sono in una zona corretta
154 INIZ:
155         call    LEGGERE
156         movf    v_STATUS, W
157         xorlw   .1
158         btfsc   STATUS, 2
159         goto    AV_GIRA_DX
160         movf    v_STATUS, W
161         xorlw   .2
162         btfsc   STATUS, 2
163         goto    AV_GIRA_DX2
```

All'inizio del programma configureremo la porta B come uscita, per poter gestire i motori. Prima di arrivare al ciclo principale troviamo una funzione di inizializzazione, che utilizza una variabile della memoria EEPROM del PIC.
Questa memoria EEPROM ha la caratteristica di non essere volatile, in quanto il dato inserito si mantiene anche quando si toglie l'alimentazione al robot.
Scriveremo in questa memoria l'ultimo stato delle zampe del robot prima di fermarsi, poiché ogni volta che si inizia il programma, il robot dovrà sapere in che posizione si trovano le zampe.

Modo zampe: Esapodo (I)

```
207 ;*****************************************************************************
208 ;Routine che fa camminare il robot in modo esapodo in avanti
209 CICLO:
210         OFF_ZAMPA1
211         OFF_ZAMPA2
212         OFF_ZAMPA3
213         movlw   .1
214         movwf   v_STATUS
215         call    SCRIVERE
216 ;STATUS 1
217 AV_GIRA_DX:
218         OFF_ZAMPA1
219         OFF_ZAMPA2
220         DX_ZAMPA3
221         btfss   PORTC,2
222         goto    AV_GIRA_DX
223         call    DELAY
224         btfss   PORTC,2
225         goto    AV_GIRA_DX
226         movlw   .2
227         movwf   v_STATUS
228         call    SCRIVERE
229 ;STATUS 2
230 AV_GIRA_DX2:
231         OFF_ZAMPA1
232         OFF_ZAMPA2
233         DX_ZAMPA3
234         btfsc   PORTC,2
235         goto    AV_GIRA_DX2
236         call    DELAY
```

Qui possiamo vedere l'inizio del ciclo principale del programma, che ha il compito di generare un movimento di avanzamento continuo del robot nella configurazione con le zampe. Per realizzare questo movimento di avanzamento, dobbiamo eseguire i passi già spiegati nella meccanica di Pathfinder in modo zampe. Ogni volta che eseguiamo un nuovo movimento delle zampe, lo stato attuale delle stesse viene scritto nella prima posizione della memoria EEPROM, per fare in modo che il robot all'inizio del programma possa conoscere l'ultimo stato all'interno della sequenza del movimento in cui si trovava.

```
386 ;*****************************************************
387 ;Routine per la scrittura nella memoria EEPROM.
388 ;Si scrive all'indirizzo 0 il valore della variabile v_STATUS
389 SCRIVERE:
390         bcf     STATUS, 5
391         bsf     STATUS, 6       ;Banco 2
392         movlw   0x00
393         movwf   EEADR
394         bcf     STATUS, 6
395         movf    v_STATUS, w
396         bsf     STATUS, 6
397         movwf   EEDATA
398         bsf     STATUS, 5
399         bcf     EECON1, EEPGD
400         bsf     EECON1, WREN
401         bcf     INTCON, GIE
402         movlw   0x55
403         movwf   EECON2
404         movlw   0xAA
405         movwf   EECON2
406         bsf     EECON1, WR
407         bcf     STATUS, 5
408         bcf     STATUS, 6
409         return
410 ;*****************************************************************************
411 ;LEGGERE: Routine che legge l'indirizzo 0 della memoria EEPROM e lascia il r
412 ;nella variabile v_STATUS
413 LEGGERE:
414         bsf     STATUS, 6
415         bcf     STATUS, 5       ;Banco 2
416         movlw   .0
417         movwf   EEADR
418         bsf     STATUS, 5       ;Banco 3
```

Nella parte finale del programma si trovano le routines che servono per scrivere e leggere la memoria EEPROM del microcontroller. Si tratta di due funzioni generiche che possiamo utilizzare in qualsiasi programma e che scrivono nella posizione 0 della memoria EEPROM il contenuto della variabile della memoria RAM, chiamata v_STATUS. A differenza della memoria RAM la cui scrittura è immediata, la memoria EEPROM impiega 10 ms per scrivere un dato, per questo ci dobbiamo assicurare che tra scrittura e scrittura trascorra un certo tempo. Nel caso specifico di questo programma, fra i diversi movimenti delle zampe del robot si supera sempre questo minimo tempo di scrittura.

In questa immagine possiamo vedere Pathfinder configurato in modo esapodo, mentre realizza un movimento di avanzamento con le zampe. Per fare questo dobbiamo scrivere il contenuto del file esap1.hex nella Smartcard del robot e inserirla nella scheda di alimentazione. Dopo qualche secondo il robot inizierà l'esecuzione del programma avanzando in modo esapodo. La prima volta che eseguiremo il programma, le zampe dovranno essere perfettamente centrate e i sensori ottici dovranno vedere la banda nera degli encoder, altrimenti i movimenti che eseguirà il robot non saranno corretti.