Programmazione di Digital Signal Controller (dsPIC33F):

Si descriva la configurazione degli SFR e la programmazione della Interrupt Service Routine di un dsPIC33F per l'applicazione con le seguenti caratteristiche tecniche:

1. L'oscillatore del dsPIC33F è stato configurato (a priori) per avere frequenza di istruzione 30 MIPS (Fcy = 30 MHz).

2. Si prevede di regolare la velocità di un motore elettrico tramite un convertitore di potenza unidirezionale, comandato unicamente dal pin PWM2L1 del modulo Motor Control PWM2 (McPWM2). La frequenza della modulazione PWM deve essere di 8 KHz. Si ricordano nel seguito le formule per ottenere tale frequenza, a seconda che il timer del modulo McPWM2 sia impostato in Free Running o in Up/Down Counting Mode:

14-1: PWM Period Calculation for Free Running Count Mode (PTMOD = 00 or 01)

$$PxTPER = \frac{FCY}{FPWM \times (PxTMR\ Prescaler)} - 1$$

14-2: PWM Period Calculation in Up/Down Counting Modes (PTMOD = 10 or 11)

$$PxTPER = \frac{FCY}{FPWM \times (PxTMR\ Prescaler) \times 2} - 1$$

3. La velocità del motore è misurata tramite una dinamo tachimetrica, la cui uscita è una tensione direttamente proporzionale alla velocità di rotazione con costante:
   Kv = 0,00125 V/RPM

4. La velocità massima dei motori utilizzati può variare leggermente da unità ad unità, ma è sempre inferiore a 2050 RPM.

5. L'uscita della dinamo è collegata al segnale analogico AN4, del quale è richiesta la conversione in digitale a 12 bit ed il campionamento sincronizzato con il periodo del segnale PWM (**N.B.** vedere SSRC, Sample Source Clock Select, per ADC e P2SECMP, Special Event Compare SFR per McPWM2).

6. Per ottimizzare le risorse del dsPIC, si vuole impostare la conversione A/D utilizzando una tensione di riferimento esterna collegata al pin 2 (VREF+), tale per cui la messa in scala della misura di velocità (in RPM) possa essere fatta utilizzando solamente aritmetica intera e senza uso di operazioni di divisione. Per generare tale tensione esterna, sono a disposizione i seguenti circuiti integrati:
   - Maxim MAX6610: Vref = 2,56 V fisso
   - Texas Instruments LM4040: Vref selezionabile tra 2,048 V; 2,5 V; 3 V; 4,096 V; 5 V; 8,192 V; 10 V.

RISPOSTA:

Valori di configurazione degli SFR:

```
////ADC CONFIG
    //Pin RB2 (AN4) Tristate as INPUT
    TRISBbits.TRISB2 = 1;
    // Config analog pins
    // all digital..
    AD1PCFGL = 0xFFFF;
    //with one exception (AN4)
    AD1PCFGLbits.PCFG4 = 0;

    // Initialize MUXA Input Selection
    AD1CHS0bits.CH0SA = 4; // Select AN3 for CH0 +ve input
    AD1CHS0bits.CH0NA = 0; // Select VREF- for CH0 -ve input

    // Set External VREF+ as ADVREF+
    AD1CON2bits.VCFG = 1;

    // 12 Bit ADC
    AD1CON1bits.AD12B = 1;

    // ADC trigger (Sample Source Select bits)
    AD1CON1bits.SSRC = 5; //McPWM2

    // ENABLE Auto-sample
    AD1CON1bits.ASAM = 1;

    // Power ON converter
    AD1CON1bits.ADON = 1;

    // ADC Interrupt Enabled (flag is set when ADC is done!)
    IEC0bits.AD1IE = 1;

//// McPWM2 config
    // Timebase OFF (turned on later), no pre/post-scaler, free-running
    P2TCON = 0x0000;
    P2TPER = 3749; // Period register for 8 KHz

    PWM2CON1bits.PMOD1 = 1; // Independent PWM pairs (PWM2H1/L1)
    PWM2CON1bits.PEN1L = 1; // PWM2L1 as PWM output

    P2SECMPbits.SEVTDIR = 0; // Special event on up-counting
    P2SECMPbits.SEVTCMP = 3749; // Special event at the end of PWM period
                                // (Not really important in this case..)

    P2DC1 = 0; // Zero-duty on PWM2L1

    P2TCONbits.PTEN = 1; // enable the PWM generator
```

```
#include <stdint.h>

Int16_t Velocity_RPM;

void __attribute__((interrupt,no_auto_psv)) _ADC1Interrupt(void)
{
// READ ADC CONVERSION RESULT AND SCALE VELOCITY MEASURE

// ASSUMPTION:
// 1) Max velocity is APPROXIMATED AS 2048 RPM
//    => Dynamo Vout max = 2048 * Kv =2,56 V
// 2) VREF+ is selected as 2,56 V (Maxim MAX6610)
// 3) Maximum ADC output is 2^12 = 4096 with 2,56 Vin on AN4
//    => this maximum value corresponds to 2048 RPM, i.e.:
//       ADCout = Kin * Velocity_RPM with Kin = 4096 / 2,56[V] . 2,56[V] / 2048[RPM] = 2
// 4) THEREFORE Velocity_RPM = ADC output / 2
//    (or equivalently right-shifted by 1)

Velocity_RPM = ADC1BUF0 >> 1;

// DON'T FORGET TO RESET THE INTERRUPT FLAG BEFORE EXIT!!!
    IFS0bits.AD1IF = 0;

} // END ADC1Interrupt ISR
```