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 vuole acquisire la misura di quattro termocoppie di tipo K, tutte opportunamente condizionate in modo tale da ottenere la seguente messa in scala del segnale analogico per l'acquisizione della temperatura:
   - Range di misura: 0 – 400 °C
   - Range di tensione in ingresso al dsPIC33F: 0,5 – 3 V

3. I segnali condizionati delle quattro termocoppie sono collegati rispettivamente agli input analogici AN0, AN3, AN4 e AN5. Si desidera effettuare il campionamento simultaneo dei quattro input con risoluzione della conversione a 10 bit, ed in modo che il tempo di campionamento sia fissato a 25 ms.
   NOTA BENE: Si suppone di utilizzare un dsPIC33F non dotato di modulo DMA, nel quale è consigliabile impostare il convertitore in modo che generi un interrupt ogni quattro cicli di conversione (v. campo SMPI in AD1CON2). Si noti inoltre che con questa modalità i risultati della conversione dei segnali assegnati ai sample/hold CH0, CH1, CH2 e CH3 sono memorizzati rispettivamente nei registri ADC1BUF0 e ADC1BUF1, ADC1BUF2 e ADC1BUF3.

4. Si deve effettuare la messa in scala delle temperature rilevate dalle termocoppie, espresse in °C, utilizzando solamente aritmetica intera (Fixed-Point).

5. L'alimentazione del dsPIC33F è a 3,3 V e non sono previste alternative all'uso di tale tensione come V_{REFH} per il convertitore A/D (e V_{REFL} a 0 V).

Suggerimenti:

- Utilizzare il Timer3 a 16 bit con un opportuno valore di prescaler oppure la concatenazione Timer2/Timer3 a 32 bit come Sample Clock Source Select per l'ADC.
- Determinare l'operazione di messa in scala dalla caratteristica ingresso/uscita delle termocoppie, al fine di ottenere come risultato le temperature in °C con formato Fixed-Point Q15 (i.e. valori reali moltiplicati per 2^{15} e arrotondati all'intero), ma con una variabile intera a 32 bit con segno.
- Si noti che per il campionamento simultaneo degli input AN0, AN3, AN4 e AN5 esiste una unica configurazione possibile per i sample/hold CH0, CH1, CH2 e CH3. Si noti inoltre che il modulo prevede una configurazione "alternata" Sample A / Sample B di tali canali che può essere scambiata automaticamente. Tale funzionalità non è richiesta e la configurazione dei canali CH0 (tramite registro AD1CHS0) e CH1/CH2/CH3 (tramite registro AD1CHS123) può essere impostata per il solo Sample A.

RISPOSTA:

Valori di configurazione degli SFR:

```
////ADC CONFIG
    //Pins RA0 (AN0), RB1 (AN3), RB2 (AN4), RB3 (AN5) Tristate as INPUTs
    TRISAbits.TRISA0 = 1;
    TRISBbits.TRISB1 = 1;
    TRISBbits.TRISB2 = 1;
    TRISBbits.TRISB3 = 1;
    // Config AN0, AN3, AN4 and AN5 as analog pins
    AD1PCFGLbits.PCFG0 = 0;
    AD1PCFGLbits.PCFG3 = 0;
    AD1PCFGLbits.PCFG4 = 0;
    AD1PCFGLbits.PCFG5 = 0;


    // Initialize MUXA Input Selection for CH0..
    AD1CHS0bits.CH0SA = 0; // Select AN0 for CH0 +ve input
    AD1CHS0bits.CH0NA = 0; // Select VREF- for CH0 -ve input
    // .. and CH1
    AD1CHS123bits.CH123SA = 1; // Select AN3-4-5 for CH1-2-3 +ve inputs
    AD1CHS123bits.CH123NA = 0; // Select VREF- for CH0 -ve input


    // 10 Bit ADC
    AD1CON1bits.AD12B = 0;
    // Integer format
    AD1CON1bits.FORM = 0;
    // ADC trigger (Sample Source Select bits)
    AD1CON1bits.SSRC = 2; //Timer 3
    // SIMULTANEOUS SAMPLING
    AD1CON1bits.SIMSAM = 1;
    // ENABLE Auto-sample
    AD1CON1bits.ASAM = 1;

    // VOLTAGE REFERENCES: Avdd (3,3V) / Avss (0v)
    AD1CON2bits.VCFG = 0;
    // CONVERT CH0, CH1, CH2 and CH3
    AD1CON2bits.CHPS = 2;
    // Interrupt every 4 conversions
    AD1CON2bits.SMPI = 3; // = 0 means every conversion, = 1 every 2 etc

    // Power ON converter
    AD1CON1bits.ADON = 1;

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

```
//// TIMER2/3 config (32 bit, no prescaler)
//// NOTE: The 32 bit timer is controlled by T2CON but generates
//// interrupts from Timer3
    T2CONbits.T32 = 1; // 32 bit mode
    T2CONbits.TON = 0; // ALL other details are in T3CON
    T2CONbits.TCS = 0; // Internal clock source
    T2CONbits.TGATE = 0; // NO gate control
    T2CONbits.TCKPS = 0; // NO Prescaler
    TMR3 = 0;
    TMR2 = 0;
    // 25 ms period @ 30 MIPS: 750.000 = 0xB71B0, split on TMR3:TMR2
    PR3 = 0x000B; // Most Significative Word of the 32 bit timer
    PR2 = 0x71B0; // Least Significative Word of the 32 bit timer
    // RESET FLAG, BUT DO NOT ENABLE INTERRUPT (using only ADC interrupt)
    IFS0bits.T3IF = 0;
    IEC0bits.T3IE = 0;
    T2CONbits.TON = 1; //START Timer
```

```
#include <stdint.h>

int32_t Temp0_Q15, Temp3_Q15, Temp4_Q15, Temp5_Q15;

void __attribute__((interrupt,no_auto_psv)) _ADC1Interrupt(void)
{
// READ ADC CONVERSION RESULTS AND SCALE TEMPERATURES

// Kadc = 2^10 / 3,3 [V] = 1024 / 3,3 [V] = 310,3
//
// ADC Value = Vout * Kadc
//
// Ksens = (3 – 0,5) / 400 = 0,00625 [V/°C]
//
// Vout of the sensor is TempX * 0,00625 [V/°C] + 0,5 [V]
//
// TempX = (ADC Value – 0,5 [V]*Kadc) * [ 1 / (Kadc*Ksens) ]
//
// Offset = 0,5 [V] * Kadc = 155 (half scale!)
//
// [ 1 / ( Kadc*Ksens) ] = 1,939375 ==> Q15 (i.e. * 2^15) ==> 63549

Temp0_Q15 = (int32_t) (ADC1BUF0 – 155) * 63549;
Temp3_Q15 = (int32_t) (ADC1BUF1 – 155) * 63549;
Temp4_Q15 = (int32_t) (ADC1BUF2 – 155) * 63549;
Temp5_Q15 = (int32_t) (ADC1BUF3 – 155) * 63549;


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

} // END ADC1Interrupt ISR
```