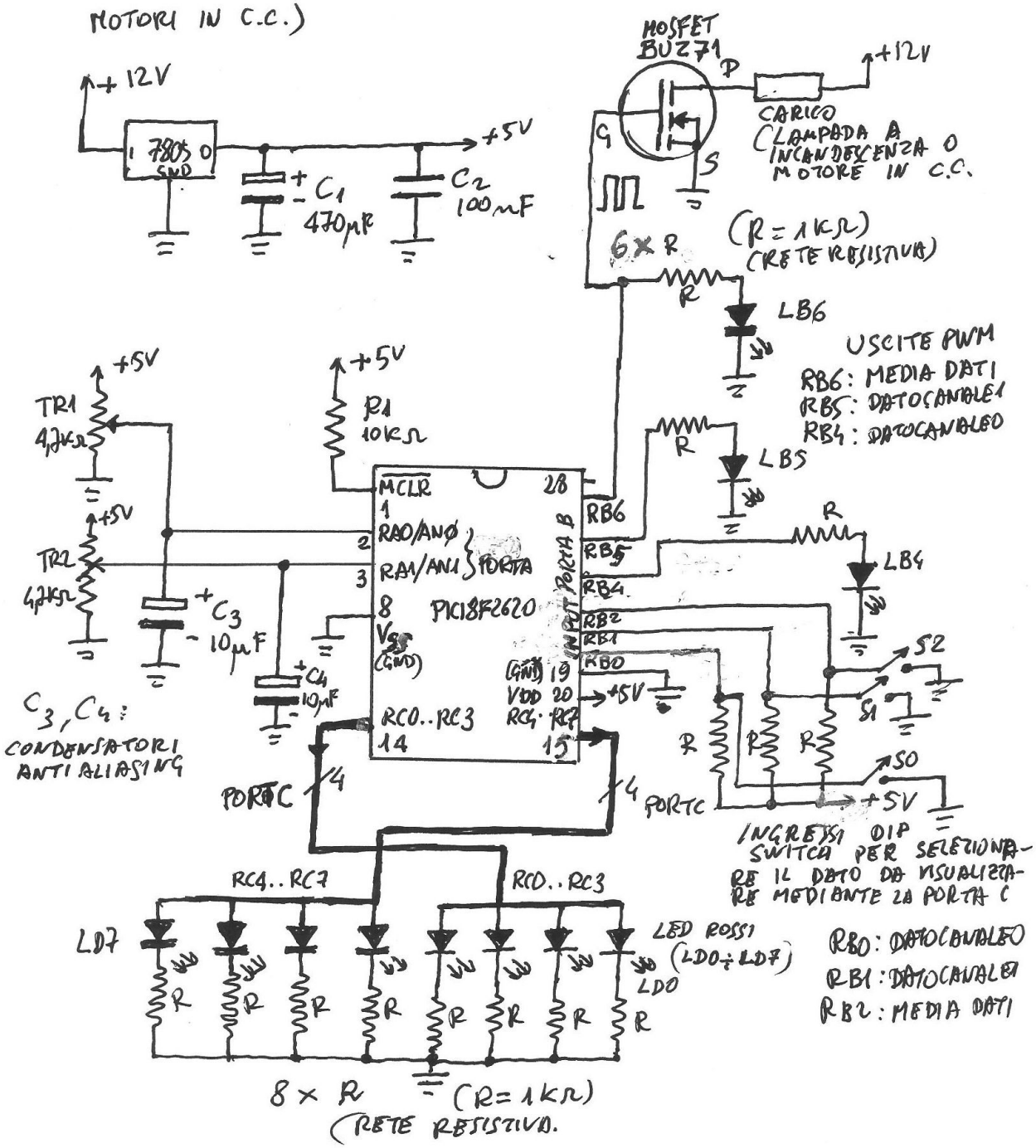
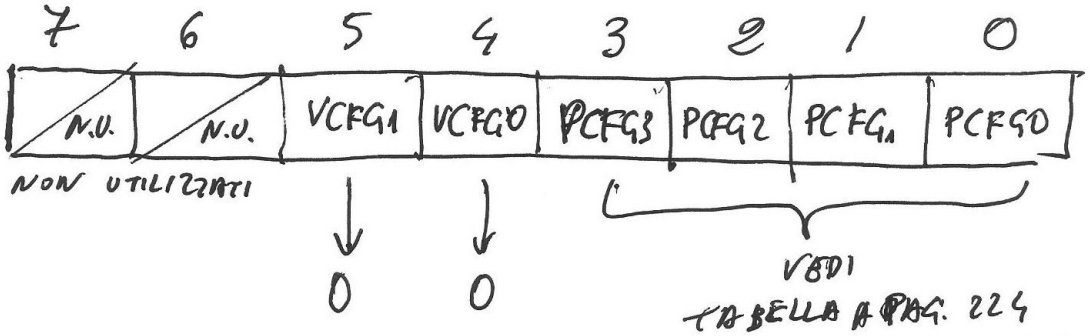
SISTEMA DI ACQUISIZIONE DATI A 2 CANALI CON USCITE PWM PER IL PILOTAGGIO DI ATTUATORI (RELÈ, MOTORI IN C.C.)

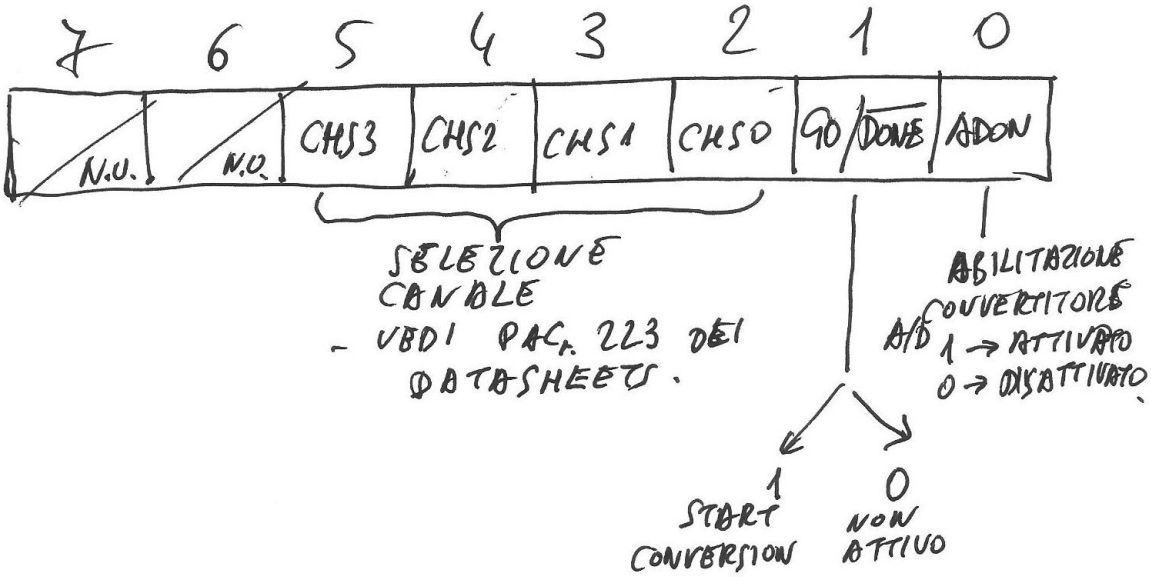


PROGRAMMAZIONE DEI REGISTRI DEL CONVERTITORE A/D INCLUSI NEL μC 18F2620

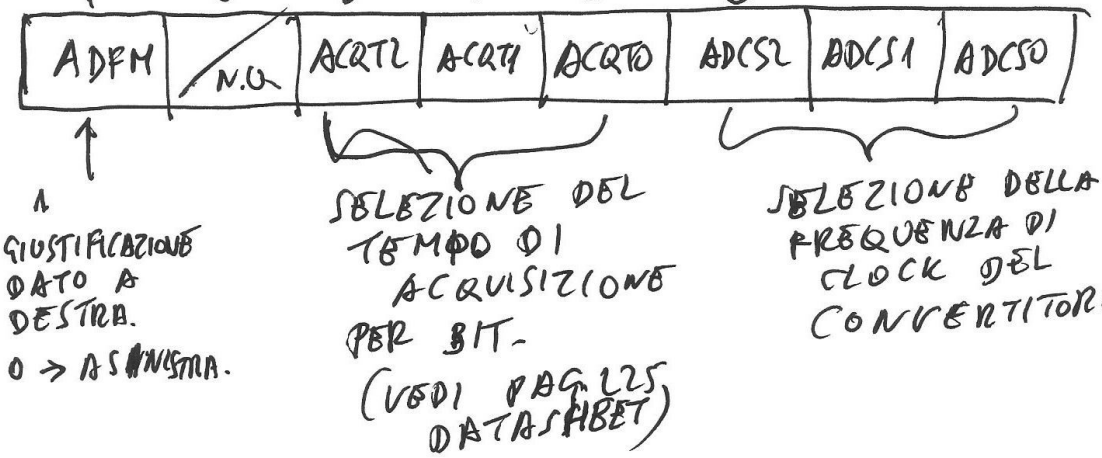
REGISTRO DI CONTROLLO ADCON1



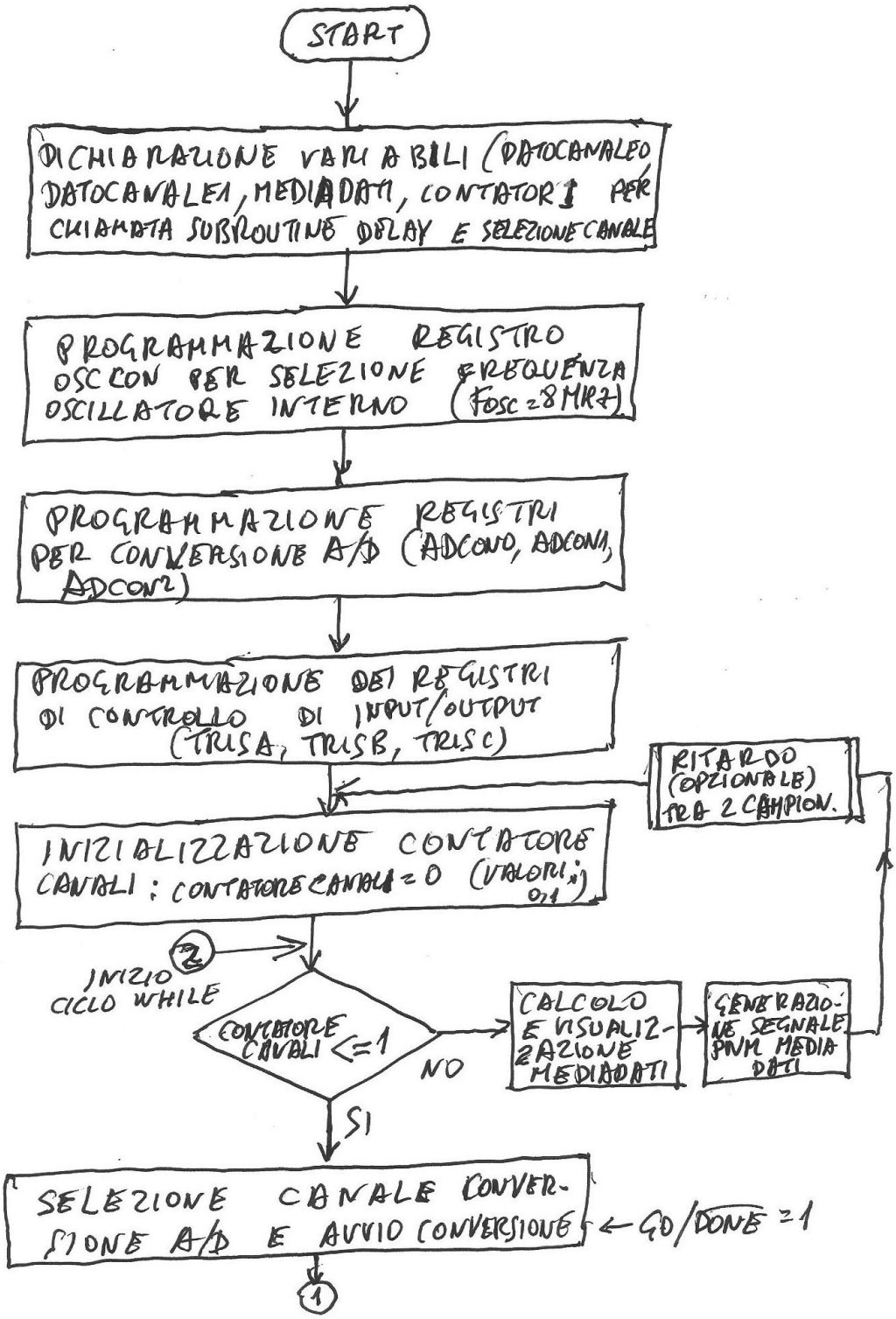
REGISTRO DI CONTROLLO ADCON0

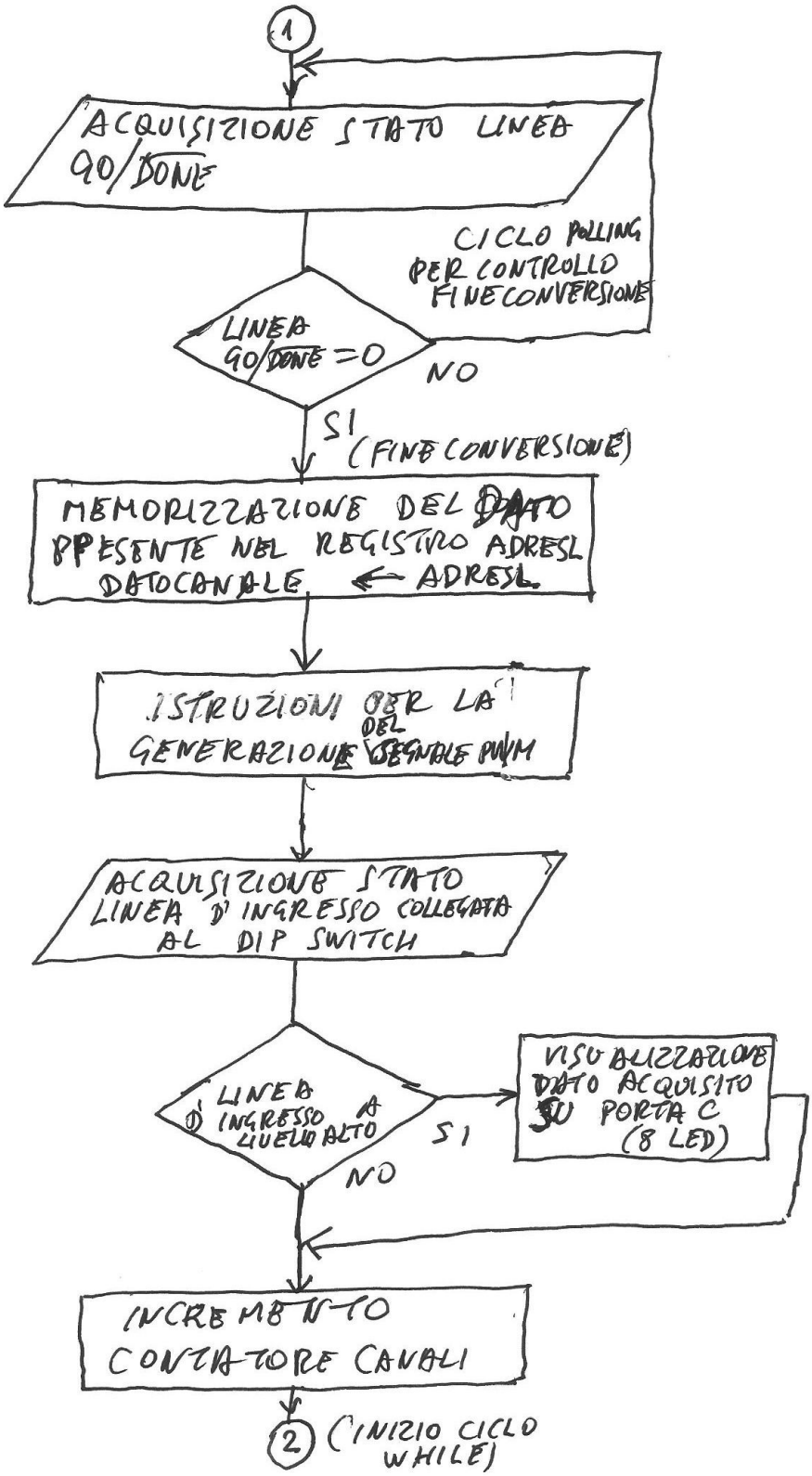


REGISTRO DI CONTROLLO ADCON2 3



FLOW CHART PER ACQUISIZIONE DATI A 2 CANALI, CON USCITE PWM




1
ACQUISIZIONE STATO LINEA GO/DONE
CICLO POLLING PER CONTROLLO FINE CONVERSIONE
LINEA GO/DONE = 0
NO
SI (FINE CONVERSIONE)
MEMORIZZAZIONE DEL DATO PRESENTE NEL REGISTRO ADRESL
DATO CANALE ← ADRESL
ISTRUZIONI PER LA GENERAZIONE DEL SEGNALE PWM
ACQUISIZIONE STATO LINEA D' INGRESSO COLLEGATA AL DIP SWITCH
LINEA D' INGRESSO A LIVELLO ALTO
SI
VISUALIZZAZIONE DATO ACQUISITO SU PORTA C (8 LED)
NO
INCREMENTO CONTATORE CANALI
2 (INIZIO CICLO WHILE)

```
/*acquisizione-dati-2canali-ad-da-pwm.c  ACQUISIZIONE DATI E CONTROLLO PWM A 2 CANALI  (CON 2 SENSORI E 2 USCITE IN PWM)*/

/** C O N F I G U R A T I O N B I T S ******************************/
/* Direttive #pragma per il compilatore C */
#pragma config OSC = INTIO7 ,IESO=OFF  /* abilitazione dell'oscillatore interno per la generazione del segnale di clock , Fosc = 8 MHz */
#pragma config LVP = OFF /*disattivazione dell' ingresso di programmazione PGM/RB5 (PGM In Circuit Low Voltage Programming) ed abilitazione della linea RB5 della*/
/*porta B come ingresso/uscita*/


/** I N C L U D E S **************************************************/

#include "p18f2620.h" /*inclusione del file caratteristico del microcontrollore impiegato PC18F2620*/
#include "math.h"
#include "delays.h"

unsigned char datocanale0, datocanale1;
unsigned int counter;
float  mediadati;
void main(void)
{
  /*programmazione del registro di controllo dell'oscillatore (OSCCON) che genera il segnale di clock*/
  OSCCONbits.SCS0=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.SCS1=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.IRCF2=1;/*1:bit di selezione della frequenza dell'oscillatore interno  Fosc = 8 MHz*/
  OSCCONbits.IRCF1=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/
  OSCCONbits.IRCF0=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/

  /*programmazione del registro di selezione (ADCON1) delle linee A/D (ingressi analogici/ingressi-uscite digitali)*/


  ADCON0bits.ADON = 1; /*abilitazione del convertitore A/D*/

  ADCON1bits.VCFG1 = 0; /*bit di configurazione della tensione di riferimento Vref- = Vss = 0 (GND)*/
  ADCON1bits.VCFG0 = 0; /* bit di configurazione della tensione di riferimento Vref+ = Vdd = +5V */
  ADCON1bits.PCFG3=1;  /*la combinazione PCGF3 = 1, PCGF2 = 1, PCGF1 = 0, PCGF0 = 1 predispone come ingressi-uscite digitali
                le linee AN2,AN3,AN4,AN8,AN9,AN10,AN11,AN12 e come ingressi analogici AN0 e AN1*/
  ADCON1bits.PCFG2=1;
  ADCON1bits.PCFG1=0;
  ADCON1bits.PCFG0=1;
```

```
  ADCON2bits.ADFM = 1; /*codice del risultato della conversione A/D giustificato a destra*/
  ADCON2bits.ACQT2 = 0;/*selezione del tempo di acquisizione riferito a 1 bit: 001 = 2 Tad, con
Tad (>0,7E-6 s ,periodo del clock di conversione) = 8 Tosc = 8/Fosc = 8/8E6 = 1E-6 s
                  Tconv = 11 Tad (per 10 bit) = 11 E-6 s ;  Fcamp = 1/Tconv = 1/11E-6s = 90909 Hz =
90,9 KHz */

  ADCON2bits.ACQT1 = 0;
  ADCON2bits.ACQT0 = 1;
  ADCON2bits.ADCS2 = 0; /*selezione della frequenza del clock di conversione: 001 ****   Fconv
= 1/Tad = Fosc/8 = 8 MHz/8 = 1 MHz*/
  ADCON2bits.ADCS1 = 0;
  ADCON2bits.ADCS0 = 1;

  TRISA = 0b11111111; /*(oppure 0xFF) predisposizione delle linee di input/output della porta A
come ingressi*/

  TRISB = 0b00001111; /*(oppure 0x00) predisposizione delle linee di input/output
RB4,RB5,RB6,RB7 della porta B come uscite (pwm),
                  da collegare ai LED e delle linee RB0,RB1,RB2,RB3 come ingressi,da collegare al
dip switch per la selezione del dato
                  (canale 0, canale 1, canale 2 o media) da visualizzare con i LED della porta C */
  TRISC = 0b00000000; /*(oppure 0x00) predisposizione di tutte le linee di input/output della porta
C come uscite,
                  da collegare ai LED*/
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  LATB = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta B */
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */


  for(;;) /*ciclo for infinito*/
    {
          ADCON0bits.CHS3=0;
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=0;
          ADCON0bits.GO = 1;   /*bit di stato GO/DONE del processo di conversione A/D*/
    while (ADCON0bits.GO == 1){};
    datocanale0 = ADRESL;
    if (PORTBbits.RB0 ==1) PORTC=datocanale0; /*visualizzazione del dato del canale 0
mediante i LED della porta C*/
    PORTBbits.RB5=1; /* generazione del segnale pwm all'uscita RB5*/
    for(counter=1; counter<= 10; counter++)Delay10TCYx(ADRESL);
    PORTBbits.RB5=0;
    for(counter=1; counter<= 5; counter++)Delay10TCYx(255-ADRESL);

    ADCON0bits.CHS3=0;
    ADCON0bits.CHS2=0;
    ADCON0bits.CHS1=0;
    ADCON0bits.CHS0=1;
```

```
    ADCON0bits.GO = 1;   /*bit di stato GO/DONE del processo di conversione A/D*/
    while (ADCON0bits.GO == 1){};
    datocanale1 = ADRESL;
    if (PORTBbits.RB1 ==1) PORTC=datocanale1;/ *visualizzazione del  dato del canale 1
mediante i LED della porta C*/
    PORTBbits.RB6 =1;  /* generazione del segnale pwm all'uscita RB6*/
    for(counter=1; counter <=10; counter++) Delay10TCYx(ADRESL);
    PORTBbits.RB6=0;
    for(counter=1; counter<= 5; counter++)Delay10TCYx(255-ADRESL);



    mediadati = (datocanale0 + datocanale1)/2.0 ; /* media dei due segnali acquisiti*/
    if (PORTBbits.RB2 ==1) PORTC=(char)mediadati;
    PORTBbits.RB4 =1;  /* generazione del segnale pwm all'uscita RB4*/
    for(counter=1; counter <=10; counter++) Delay10TCYx((char)mediadati);
    PORTBbits.RB4=0;
    for(counter=1; counter <=5; counter++) Delay10TCYx(255-(char)mediadati);
    }

}
```

CASO PARTICOLARE DI ACQUISIZIONE DATI CON UN UNICO CANALE (CON UN UNICO SENSORE)

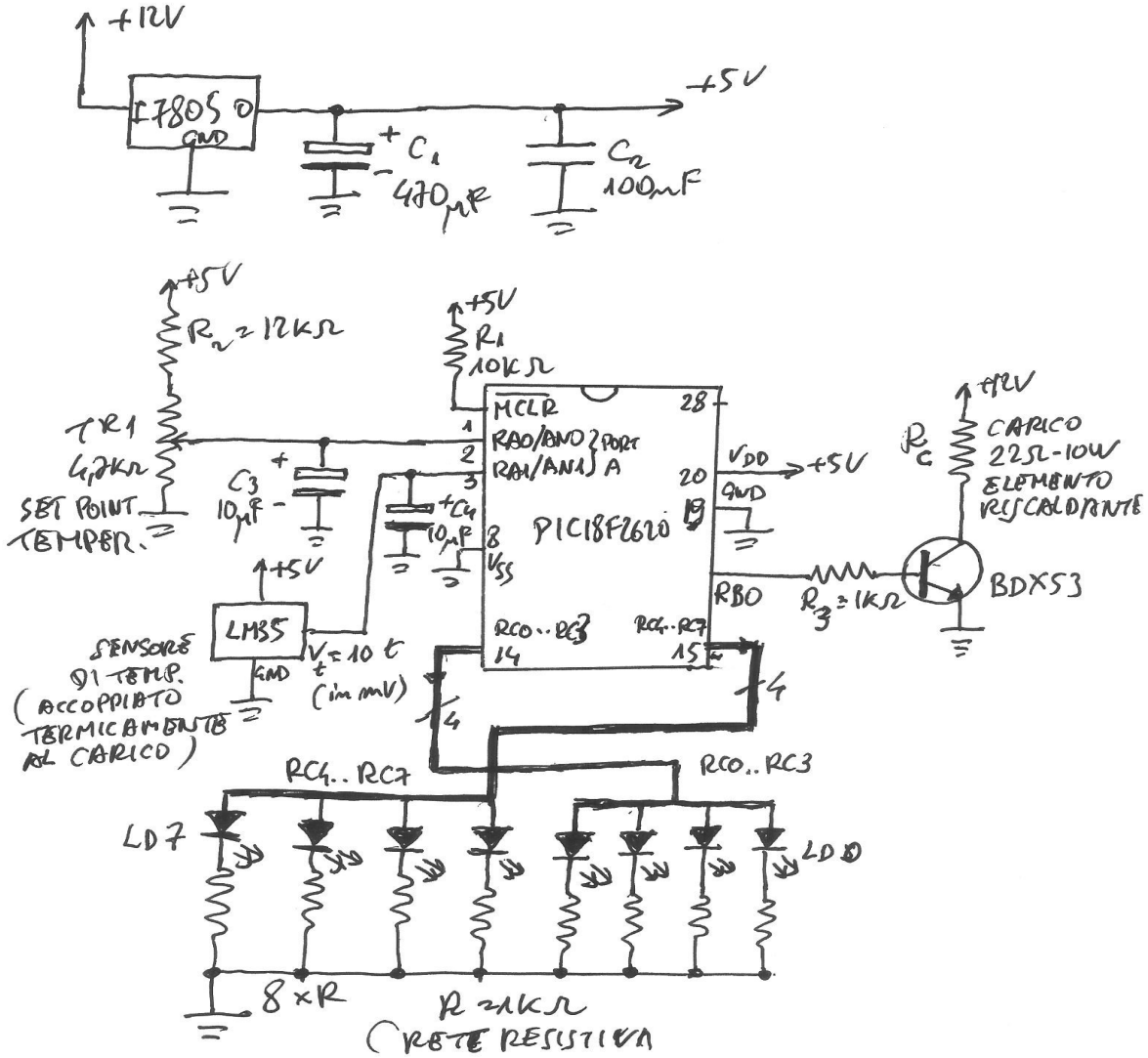
```
/*acquisizione-dati-1canale.c  ACQUISIZIONE DATI - 1 CANALE */

/** C O N F I G U R A T I O N B I T S ******************************/
/* Direttive #pragma per il compilatore C */
#pragma config OSC = INTIO7 ,IESO=OFF  /* abilitazione dell'oscillatore interno per la
generazione del segnale di clock , Fosc = 8 MHz */
#pragma config LVP = OFF /*disattivazione dell' ingresso di programmazione PGM/RB5 (PGM In
Circuit Low Voltage Programming) ed abilitazione della linea RB5 della*/
/*porta B come ingresso/uscita*/


/** I N C L U D E S **************************************************/

#include "p18f2620.h" /*inclusione del file caratteristico del microcontrollore impiegato
PC18F2620*/


void main(void)
{
  /*programmazione del registro di controllo dell'oscillatore (OSCCON) che genera il segnale di
clock*/
  OSCCONbits.SCS0=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.SCS1=1; /*1:bit di selezione dell'oscillatore interno*/
```

```
  OSCCONbits.IRCF2=1;/*1:bit di selezione della frequenza dell'oscillatore interno  Fosc = 8
MHz*/
  OSCCONbits.IRCF1=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/
  OSCCONbits.IRCF0=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/

  /*programmazione del registro di selezione (ADCON1) delle linee A/D (ingressi
analogici/ingressi-uscite digitali)*/

  ADCON0bits.CHS3 = 0; /*selezione del canale di acquisizione AN0*/
  ADCON0bits.CHS2 = 0;
  ADCON0bits.CHS1 = 0;
  ADCON0bits.CHS0 = 0;
  ADCON0bits.GO = 1; /*bit di stato del processo di conversione A/D*/
  ADCON0bits.ADON = 1; /*abilitazione del convertitore A/D*/

  ADCON1bits.VCFG1 = 0; /*bit di configurazione della tensione di riferimento Vref- = Vss = 0
(GND)*/
  ADCON1bits.VCFG0 = 0; /* bit di configurazione della tensione di riferimento Vref+ = Vdd =
+5V */
  ADCON1bits.PCFG3=1;  /*la combinazione PCGF3 = 1, PCGF2 = 1, PCGF1 = 1, PCGF0 = 0
predispone come ingressi-uscite digitali*/
                /*le linee AN1,AN2,AN3,AN4,AN8,AN9,AN10,AN11,AN12 e come ingresso
analogico AN0*/
  ADCON1bits.PCFG2=1;
  ADCON1bits.PCFG1=1;
  ADCON1bits.PCFG0=0;

  ADCON2bits.ADFM = 1; /*codice del risultato della conversione A/D giustificato a destra*/
  ADCON2bits.ACQT2 = 0;/*selezione del tempo di acquisizione riferito a 1 bit: 001 = 2 Tad, con
Tad (>0,7E-6 s ,periodo del clock di conversione) = 8 Tosc = 8/Fosc = 8/8E6 = 1E-6 s*/
                 /* Tconv = 11 Tad (per 10 bit) = 11 E-6 s ;  Fcamp = 1/Tconv = 1/11E-6s = 90909 Hz
= 90,9 KHz */

  ADCON2bits.ACQT1 = 0;
  ADCON2bits.ACQT0 = 1;
  ADCON2bits.ADCS2 = 0; /*selezione della frequenza del clock di conversione: 001 ****   Fconv
= 1/Tad = Fosc/8 = 8 MHz/8 = 1 MHz*/
  ADCON2bits.ADCS1 = 0;
  ADCON2bits.ADCS0 = 1;


  TRISB = 0b00000000; /*(oppure 0xFF) predisposizione di tutte le linee di input/output della porta
B come uscite,
                da collegare ai LED*/
  TRISC = 0b00000000; /*(oppure 0x00) predisposizione di tutte le linee di input/output della porta
C come uscite,
                da collegare ai LED*/
                 LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  LATB = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta B */
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
```

```
while(1) /*ciclo while infinito*/
    {
     ADCON0bits.GO = 1;
     while (ADCON0bits.GO == 1){};
     PORTB = ADRESH;
     PORTC = ADRESL;
     }

}
```

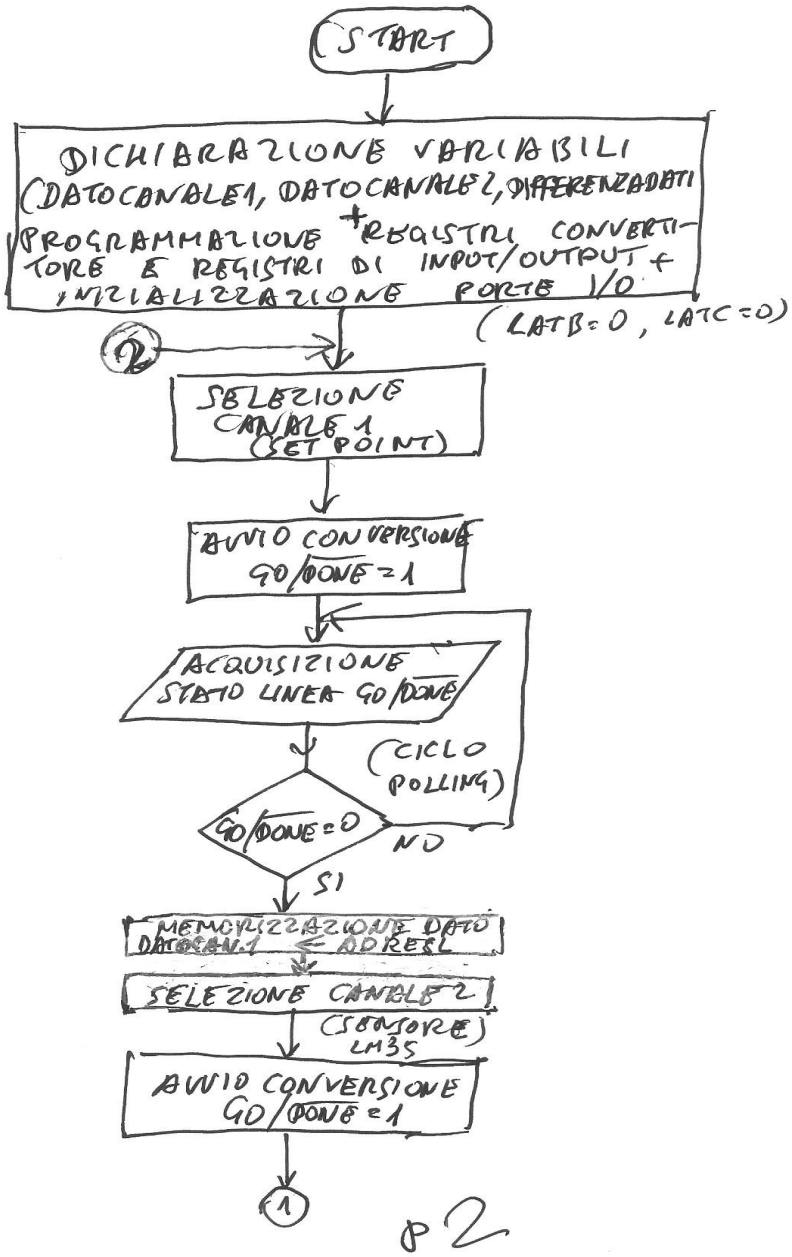
CONTROLLO AUTOMATICO DI TEMPERATURA CON PIC 18F2620

CON MODALITÀ DI FUNZIONAMENTO ON/OFF E PWM

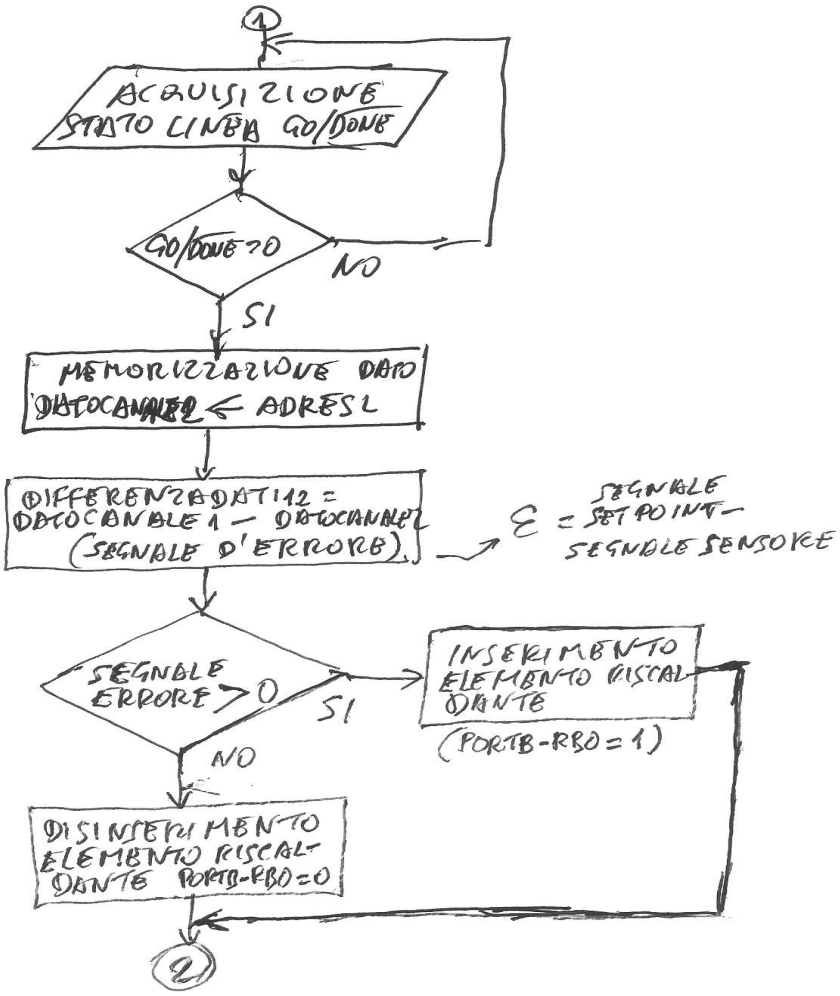


p 1

FLOW CHART PER IL CONTROLLO DI TEMPERATURA ON/OFF SENZA ISTERESI



p2

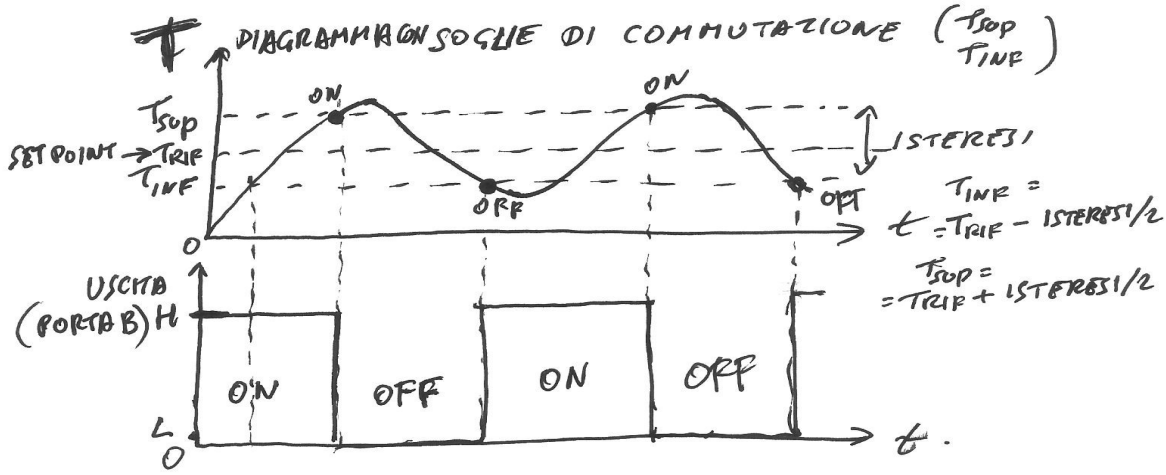

1
ACQUISIZIONE STATO LINEA GO/DONE
GO/DONE=0
NO
SI
MEMORIZZAZIONE DATO
DATOCANALE ← ADRESL
DIFFERENZADATI12 = DATOCANALE1 − DATOCANALE2 (SEGNALE D'ERRORE).
E = SEGNALE SETPOINT − SEGNALE SENSORE
SEGNALE ERRORE > 0
SI
NO
INSERIMENTO ELEMENTO RISCALDANTE
(PORTB-RB0 = 1)
DISINSERIMENTO ELEMENTO RISCALDANTE PORTB-RB0=0
2

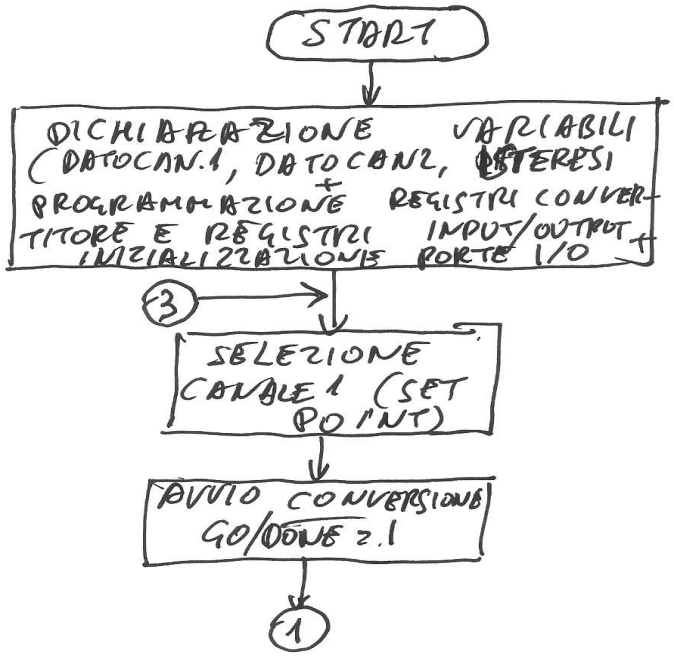
p 3

```
/*controllo-temperatura-on-off.c   controllo di temperatura on-off , con un' unica soglia di
intervento , equivalente ad un comparatore semplice*/

/** C O N F I G U R A T I O N B I T S ******************************/
/* Direttive #pragma per il compilatore C */
#pragma config OSC = INTIO7, IESO=OFF, WDT=ON /* abilitazione dell'oscillatore interno per
la generazione del segnale di clock , Fosc = 8 MHz */
#pragma config LVP = OFF /*disattivazione dell' ingresso di programmazione PGM/RB5 (PGM In
Circuit Low Voltage Programming) ed abilitazione della linea RB5 della*/
/*porta B come ingresso/uscita*/


/** I N C L U D E S **************************************************/

#include "p18f2620.h" /*inclusione del file caratteristico del microcontrollore impiegato
PC18F2620*/
#include "delays.h"
#include "math.h"

char volatile datocanale1,datocanale2,differenzadati12;

void main(void)
{
  /*programmazione del registro di controllo dell'oscillatore (OSCCON) che genera il segnale di
clock*/
  OSCCONbits.SCS0=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.SCS1=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.IRCF2=1;/*1:bit di selezione della frequenza dell'oscillatore interno  Fosc = 8
MHz*/
  OSCCONbits.IRCF1=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/
  OSCCONbits.IRCF0=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/

  /*programmazione del registro di selezione (ADCON1) delle linee A/D (ingressi
analogici/ingressi-uscite digitali)*/


  ADCON0bits.ADON = 1; /*abilitazione del convertitore A/D*/

  ADCON1bits.VCFG1 = 0; /*bit di configurazione della tensione di riferimento Vref- = Vss = 0
(GND)*/
  ADCON1bits.VCFG0 = 0; /* bit di configurazione della tensione di riferimento Vref+ = Vdd =
+5V */
  ADCON1bits.PCFG3=1;  /*la combinazione PCGF3 = 1, PCGF2 = 1, PCGF1 = 1, PCGF0 = 0
predispone come ingressi-uscite digitali*/
                /*le linee AN2,AN3,AN4,AN8,AN9,AN10,AN11,AN12 e come ingressi analogici
AN0 e AN1*/
  ADCON1bits.PCFG2=1;
  ADCON1bits.PCFG1=0;
  ADCON1bits.PCFG0=1;
```

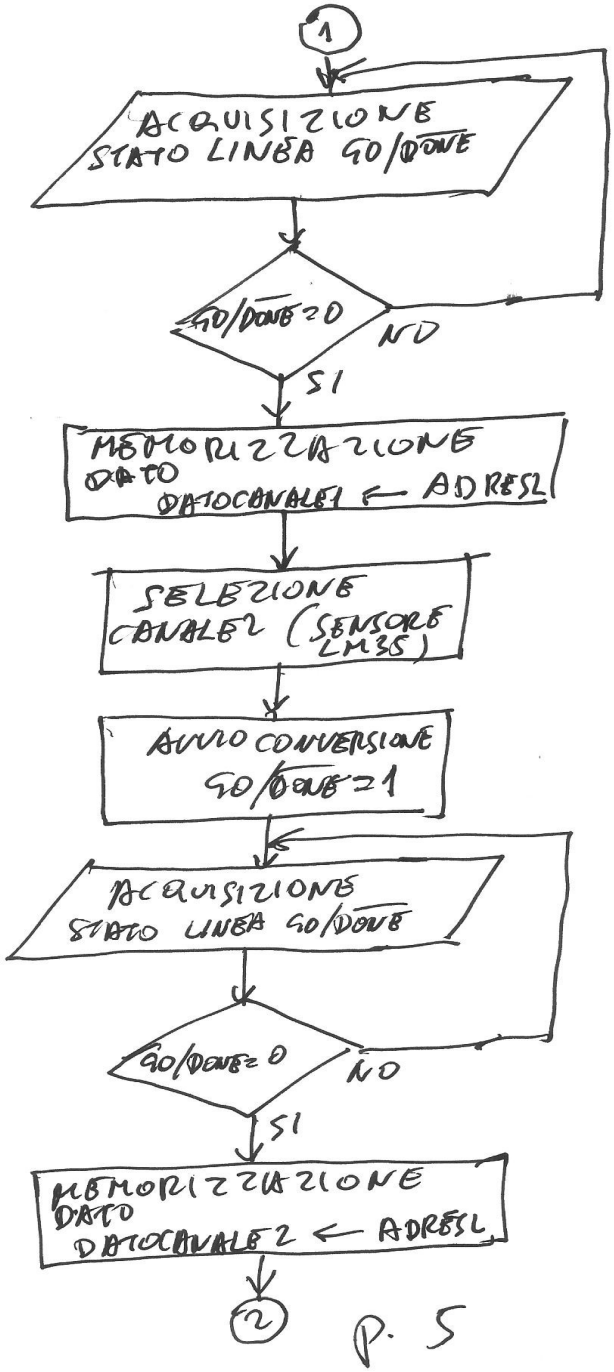
```
  ADCON2bits.ADFM = 1; /*codice del risultato della conversione A/D giustificato a destra*/
  ADCON2bits.ACQT2 = 0;/*selezione del tempo di acquisizione riferito a 1 bit: 001 = 2 Tad, con
Tad (>0,7E-6 s ,periodo del clock di conversione) = 8 Tosc = 8/Fosc = 8/8E6 = 1E-6 s*/
                    /* Tconv = 11 Tad (per 10 bit) = 11 E-6 s ;  Fcamp = 1/Tconv = 1/11E-6s = 90909 Hz
= 90,9 KHz */
  ADCON2bits.ACQT1 = 0;
  ADCON2bits.ACQT0 = 1;
  ADCON2bits.ADCS2 = 0; /*selezione della frequenza del clock di conversione: 001 ****   Fconv
= 1/Tad = Fosc/8 = 8 MHz/8 = 1 MHz*/
  ADCON2bits.ADCS1 = 0;
  ADCON2bits.ADCS0 = 1;

  TRISA = 0b00000011; /*(oppure 0xFF) predisposizione delle linee di RA1..RA7 di input/output
della porta A come uscite,e delle linee RA0 e RA1 come ingressi*/

  TRISB = 0b00000000; /*(oppure 0xFF) predisposizione di tutte le linee di input/output della porta
B come uscite,
                    da collegare ai LED*/
  TRISC = 0b00000000; /*(oppure 0x00) predisposizione di tutte le linee di input/output della porta
C come uscite,
                    da collegare ai LED*/
                     LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  LATB = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta B */
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  for(;;)
     {
          ADCON0bits.CHS3=0;
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=0;
     ADCON0bits.GO = 1;
     while (ADCON0bits.GO == 1){};
     datocanale1=ADRESL;
     ADCON0bits.CHS3=0;
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=1;
     ADCON0bits.GO = 1;
     while (ADCON0bits.GO == 1){};
     datocanale2=ADRESL;
     differenzadati12=datocanale1-datocanale2;
     LATC = differenzadati12;
     if(differenzadati12>=0)
                              {
                               PORTBbits.RB0=1;
                               }
    else PORTBbits.RB0 =0;
     }
  }
```

GRAFICI E FLOW CHART PER IL CONTROLLO DI TEMPERATURA ON/OFF CON ISTERESI

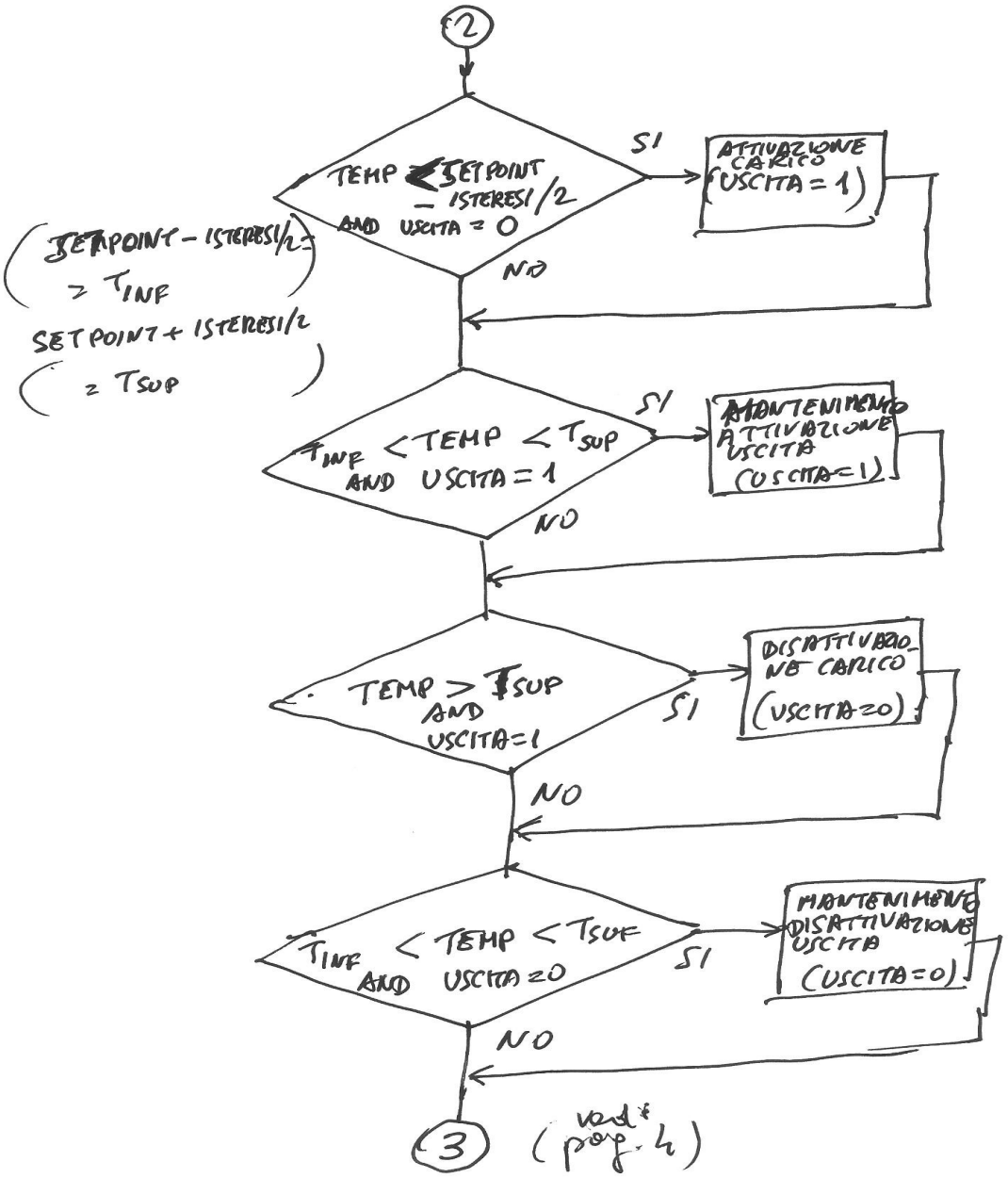




8 4


1
ACQUISIZIONE STATO LINEA GO/DONE
GO/DONE=0
NO
SI
MEMORIZZAZIONE DATO DATOCANALE1 ← ADRESL
SELEZIONE CANALE2 (SENSORE LM35)
AVVIO CONVERSIONE GO/DONE=1
ACQUISIZIONE STATO LINEA GO/DONE
GO/DONE=0
NO
SI
MEMORIZZAZIONE DATO DATOCANALE2 ← ADRESL
2

P. 5


2
TEMP < SETPOINT − ISTERESI/2 AND USCITA = 0
SI
ATTIVAZIONE CARICO (USCITA = 1)
NO
(SETPOINT − ISTERESI/2 = T_INF
SETPOINT + ISTERESI/2 = T_SUP)
T_INF < TEMP < T_SUP AND USCITA = 1
SI
MANTENIMENTO ATTIVAZIONE USCITA (USCITA=1)
NO
TEMP > T_SUP AND USCITA=1
SI
DISATTIVAZIONE CARICO (USCITA=0)
NO
T_INF < TEMP < T_SUP AND USCITA=0
SI
MANTENIMENTO DISATTIVAZIONE USCITA (USCITA=0)
NO
3
(vedi pag. 4)

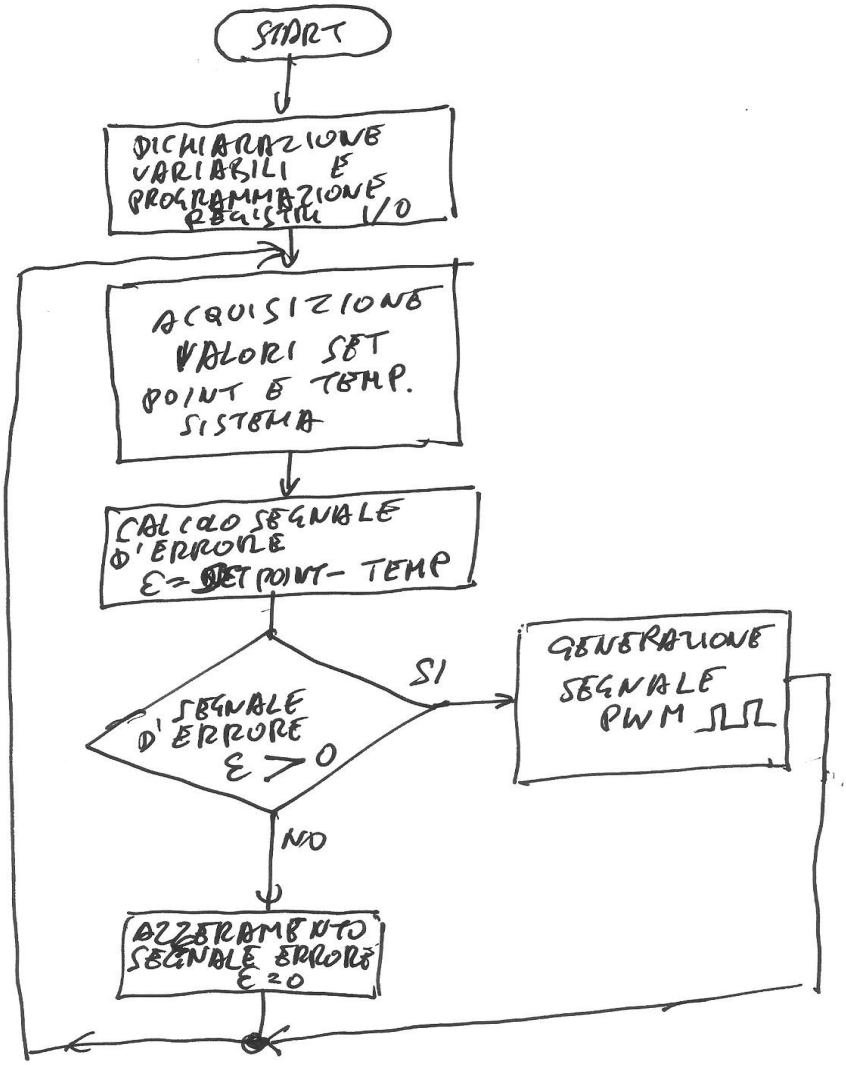
P. 6

```
/*controllo-temperatura-on-off-isteresi.c   controllo di temperatura on-off con isteresi, con 2 soglie di intervento, equivalente ad un comparatore con isteresi (trigger Scmitt)*/

/** C O N F I G U R A T I O N B I T S ******************************/
/* Direttive #pragma per il compilatore C */
#pragma config OSC = INTIO7, IESO=OFF, WDT=ON /* abilitazione dell'oscillatore interno per la generazione del segnale di clock , Fosc = 8 MHz */
#pragma config LVP = OFF /*disattivazione dell' ingresso di programmazione PGM/RB5 (PGM In Circuit Low Voltage Programming) ed abilitazione della linea RB5 della*/
/*porta B come ingresso/uscita*/


/** I N C L U D E S **************************************************/

#include "p18f2620.h" /*inclusione del file caratteristico del microcontrollore impiegato PC18F2620*/
#include "delays.h"
#include "math.h"


char volatile datocanale1,datocanale2,differenzadati12,isteresi;


void main(void)
{
  /*programmazione del registro di controllo dell'oscillatore (OSCCON) che genera il segnale di clock*/
  OSCCONbits.SCS0=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.SCS1=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.IRCF2=1;/*1:bit di selezione della frequenza dell'oscillatore interno  Fosc = 8 MHz*/
  OSCCONbits.IRCF1=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/
  OSCCONbits.IRCF0=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/

  /*programmazione del registro di selezione (ADCON1) delle linee A/D (ingressi analogici/ingressi-uscite digitali)*/


  ADCON0bits.ADON = 1; /*abilitazione del convertitore A/D*/

  ADCON1bits.VCFG1 = 0; /*bit di configurazione della tensione di riferimento Vref- = Vss = 0 (GND)*/
  ADCON1bits.VCFG0 = 0; /* bit di configurazione della tensione di riferimento Vref+ = Vdd = +5V */
  ADCON1bits.PCFG3=1;  /*la combinazione PCGF3 = 1, PCGF2 = 1, PCGF1 = 1, PCGF0 = 0 predispone come ingressi-uscite digitali*/
                       /*le linee AN2,AN3,AN4,AN8,AN9,AN10,AN11,AN12 e come ingressi analogici AN0 e AN1*/
  ADCON1bits.PCFG2=1;
  ADCON1bits.PCFG1=0;
```

```
  ADCON1bits.PCFG0=1;

  ADCON2bits.ADFM = 1; /*codice del risultato della conversione A/D giustificato a destra*/
  ADCON2bits.ACQT2 = 0;/*selezione del tempo di acquisizione riferito a 1 bit: 001 = 2 Tad, con
Tad (>0,7E-6 s ,periodo del clock di conversione) = 8 Tosc = 8/Fosc = 8/8E6 = 1E-6 s*/
                    /* Tconv = 11 Tad (per 10 bit) = 11 E-6 s ;  Fcamp = 1/Tconv = 1/11E-6s = 90909 Hz
= 90,9 KHz */

  ADCON2bits.ACQT1 = 0;
  ADCON2bits.ACQT0 = 1;
  ADCON2bits.ADCS2 = 0; /*selezione della frequenza del clock di conversione: 001 ****   Fconv
= 1/Tad = Fosc/8 = 8 MHz/8 = 1 MHz*/
  ADCON2bits.ADCS1 = 0;
  ADCON2bits.ADCS0 = 1;

  TRISA = 0b00000011; /*(oppure 0xFF) predisposizione delle linee di RA1..RA7 di input/output
della porta A come uscite,e delle linee RA0 e RA1 come ingressi*/

  TRISB = 0b00000000; /*(oppure 0xFF) predisposizione di tutte le linee di input/output della porta
B come uscite,
                    da collegare ai LED*/
  TRISC = 0b00000000; /*(oppure 0x00) predisposizione di tutte le linee di input/output della porta
C come uscite,
                    da collegare ai LED*/
                     LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  LATB = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta B */
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  isteresi = 10;
  PORTBbits.RB0 = 0;
  for(;;)
     {
          ADCON0bits.CHS3=0;    /*acquisizione del valore del set point di temperatura*/
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=0;
      ADCON0bits.GO = 1;
      while (ADCON0bits.GO == 1){};
      datocanale1=ADRESL;
      ADCON0bits.CHS3=0;  /* acquisizione della temperatura del sistema */
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=1;
      ADCON0bits.GO = 1;
      while (ADCON0bits.GO == 1){};
      datocanale2=ADRESL;
      LATC = datocanale2;
      if ((datocanale2 < datocanale1 - (char)(isteresi/2.0)) && (PORTBbits.RB0 ==0))
PORTBbits.RB0=1;
      else if ((datocanale2 >  datocanale1 - (char)(isteresi/2.0)) && (datocanale2 < datocanale1
+ (char)(isteresi/2.0))&&(PORTBbits.RB0 ==1)) PORTBbits.RB0=1;
```

```
    else if ((datocanale2 > datocanale1 + (char)(isteresi/2.0)) && (PORTBbits.RB0 ==1))
PORTBbits.RB0=0;
    else if ((datocanale2 >  datocanale1 - (char)(isteresi/2.0)) && (datocanale2 < datocanale1
+ (char)(isteresi/2.0))&&(PORTBbits.RB0 ==0)) PORTBbits.RB0=0;
    }
}
```

MODIFICA DEL FLOW CHART PER IL CONTROLLO PWM.



P. 9

```
/*controllo-temperatura-pwm.c   controllo di temperatura lineare in pwm*/

/** C O N F I G U R A T I O N B I T S ******************************/
/* Direttive #pragma per il compilatore C */
#pragma config OSC = INTIO7, IESO=OFF, WDT=ON /* abilitazione dell'oscillatore interno per la generazione del segnale di clock , Fosc = 8 MHz */
#pragma config LVP = OFF /*disattivazione dell' ingresso di programmazione PGM/RB5 (PGM In Circuit Low Voltage Programming) ed abilitazione della linea RB5 della*/
/*porta B come ingresso/uscita*/


/** I N C L U D E S **************************************************/

#include "p18f2620.h" /*inclusione del file caratteristico del microcontrollore impiegato PC18F2620*/
#include "delays.h"
#include "math.h"


char volatile datocanale1,datocanale2,segnaledierrore;


void main(void)
{
  /*programmazione del registro di controllo dell'oscillatore (OSCCON) che genera il segnale di clock*/
  OSCCONbits.SCS0=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.SCS1=1; /*1:bit di selezione dell'oscillatore interno*/
  OSCCONbits.IRCF2=1;/*1:bit di selezione della frequenza dell'oscillatore interno  Fosc = 8 MHz*/
  OSCCONbits.IRCF1=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/
  OSCCONbits.IRCF0=1;/*1:bit di selezione della frequenza dell'oscillatore interno*/

  /*programmazione del registro di selezione (ADCON1) delle linee A/D (ingressi analogici/ingressi-uscite digitali)*/


  ADCON0bits.ADON = 1; /*abilitazione del convertitore A/D*/

  ADCON1bits.VCFG1 = 0; /*bit di configurazione della tensione di riferimento Vref- = Vss = 0 (GND)*/
  ADCON1bits.VCFG0 = 0; /* bit di configurazione della tensione di riferimento Vref+ = Vdd = +5V */
  ADCON1bits.PCFG3=1;  /*la combinazione PCGF3 = 1, PCGF2 = 1, PCGF1 = 1, PCGF0 = 0 predispone come ingressi-uscite digitali*/
                  /*le linee AN2,AN3,AN4,AN8,AN9,AN10,AN11,AN12 e come ingressi analogici AN0 e AN1*/
  ADCON1bits.PCFG2=1;
  ADCON1bits.PCFG1=0;
  ADCON1bits.PCFG0=1;
```

```
  ADCON2bits.ADFM = 1; /*codice del risultato della conversione A/D giustificato a destra*/
  ADCON2bits.ACQT2 = 0;/*selezione del tempo di acquisizione riferito a 1 bit: 001 = 2 Tad, con
Tad (>0,7E-6 s ,periodo del clock di conversione) = 8 Tosc = 8/Fosc = 8/8E6 = 1E-6 s*/
                    /* Tconv = 11 Tad (per 10 bit) = 11 E-6 s ;  Fcamp = 1/Tconv = 1/11E-6s = 90909 Hz
= 90,9 KHz */

  ADCON2bits.ACQT1 = 0;
  ADCON2bits.ACQT0 = 1;
  ADCON2bits.ADCS2 = 0; /*selezione della frequenza del clock di conversione: 001 ****   Fconv
= 1/Tad = Fosc/8 = 8 MHz/8 = 1 MHz*/
  ADCON2bits.ADCS1 = 0;
  ADCON2bits.ADCS0 = 1;

  TRISA = 0b00000011; /*(oppure 0xFF) predisposizione delle linee di RA1..RA7 di input/output
della porta A come uscite,e delle linee RA0 e RA1 come ingressi*/

  TRISB = 0b00000000; /*(oppure 0xFF) predisposizione di tutte le linee di input/output della porta
B come uscite,
                    da collegare ai LED*/
  TRISC = 0b00000000; /*(oppure 0x00) predisposizione di tutte le linee di input/output della porta
C come uscite,
                    da collegare ai LED*/
                     LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  LATB = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta B */
  LATC = 0b00000000; /*(oppure 0x00) azzeramento del latch della porta C */
  PORTBbits.RB0 = 0;
  for(;;)
     {
          ADCON0bits.CHS3=0;    /*acquisizione del valore del set point di temperatura*/
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=0;
      ADCON0bits.GO = 1;
      while (ADCON0bits.GO == 1){};
      datocanale1=ADRESL;
      ADCON0bits.CHS3=0;   /*acquisizione della temperatura del sistema */
          ADCON0bits.CHS2=0;
          ADCON0bits.CHS1=0;
          ADCON0bits.CHS0=1;
      ADCON0bits.GO = 1;
      while (ADCON0bits.GO == 1){};
      datocanale2=ADRESL;
      segnaledierrore = datocanale1 - datocanale2;
      if(segnaledierrore>0)
                                          {
                                          LATC = segnaledierrore;
                                          PORTBbits.RB0 = 1; /* inserimento del carico */
                                          Delay10KTCYx(17*segnaledierrore);  /*ritardo max di
256 * 10000 TCY = 256 * 10000 * 4/Fosc = 1E7/8E6 s = 0,125*10 s = 1,25 s */
```

```
                                        PORTBbits.RB0 = 0; /*disinserimento del carico */
                                        Delay10KTCYx(255 -17*segnaledierrore);   /*valori del
guadagno rispetto al segnale d'errore:3,17,85*/
                                        }
            else segnaledierrore=0;
    }
}
```

USO DELLE SUBROUTINE DI RITARDO

Delay1TCY( ) fornisce un ritardo pari un ciclo istruzione TCY = 4 * Tclock = 4 /Fclock
Esempio: Se Fclock = 8 MHz, TCY = 4/8E6 = 0,5 E-6 s = 0,5 µs = 500 ns

Delay10TCYx (N) fornisce un ritardo pari a 10 N cicli istruzione = 10 N * TCY,
con il parametro N variabile nell 'intervallo 0<=N<= 255
Massimo ritardo : 2560 TCY

Delay100TCYx (N) fornisce un ritardo pari a 100 N cicli istruzione = 100 N * TCY,
con il parametro N variabile nell 'intervallo 0<=N<= 255
Massimo ritardo: 25600 TCY

Delay1KTCYx (N) fornisce un ritardo pari a 1000 N cicli istruzione = 1000 N TCY,
con il parametro N variabile nell 'intervallo 0<=N<= 255
Massimo ritardo: 256000 TCY

Delay10KTCYx (N) fornisce un ritardo pari a 10000 N cicli istruzione = 10000 N TCY,
con il parametro N variabile nell 'intervallo 0<=N<= 255
Massimo ritardo: 2560000 TCY

```
#ifndef __DELAYS_H
#define __DELAYS_H

/* PIC18 cycle-count delay routines.
 *
 *   Functions:
 *           Delay1TCY()
 *               Delay10TCY()  // 17Cxx only
 *               Delay10TCYx()
 *               Delay100TCYx()
 *               Delay1KTCYx()
 *               Delay10KTCYx()
 */

/* For definition of Nop() */
#include <p18cxxx.h>

/* Delay of exactly 1 Tcy */
#define Delay1TCY() Nop()

#define PARAM_SCLASS auto

/* Delay of exactly 10 Tcy */
#define Delay10TCY() Delay10TCYx(1)
```

```
/* Delay10TCYx
 * Delay multiples of 10 Tcy
 * Passing 0 (zero) results in a delay of 2560 cycles.
 * The 18Cxxx version of this function supports the full range [0,255]
 * The 17Cxxx version supports [2,255] and 0.
 */
void Delay10TCYx(PARAM_SCLASS unsigned char);

/* Delay100TCYx
 * Delay multiples of 100 Tcy
 * Passing 0 (zero) results in a delay of 25,600 cycles.
 * The full range of [0,255] is supported.
 */
void Delay100TCYx(PARAM_SCLASS unsigned char);

/* Delay1KTCYx
 * Delay multiples of 1000 Tcy
 * Passing 0 (zero) results in a delay of 256,000 cycles.
 * The full range of [0,255] is supported.
 */
void Delay1KTCYx(PARAM_SCLASS unsigned char);

/* Delay10KTCYx
 * Delay multiples of 10,000 Tcy
 * Passing 0 (zero) results in a delay of 2,560,000 cycles.
 * The full range of [0,255] is supported.
 */
void Delay10KTCYx(PARAM_SCLASS unsigned char);

#endif
```