



Ministero dell'Istruzione, dell'Università e della Ricerca

M048 - ESAME DI STATO DI ISTITUTO PROFESSIONALE

CORSO DI ORDINAMENTO

1/10

Indirizzo: TECNICO DELLE INDUSTRIE ELETTRONICHE

Tema di: SISTEMI, AUTOMAZIONE E ORGANIZZAZIONE DELLA PRODUZIONE

Il candidato svolga, a sua scelta, uno dei due temi proposti.

Tema n. 1

L'editore di un quotidiano locale che insiste sul territorio di una piccola provincia, decide di offrire ai suoi lettori alcuni nuovi servizi on-line.

In particolare:

1. vuole pubblicare su un sito appositamente registrato la versione on-line del giornale, inserendo gli articoli più importanti dell'edizione cartacea del giorno, con una eventuale fotografia inerente alla notizia;
2. vuole realizzare una web-radio attiva a partire dallo stesso sito, per la diffusione di notizie, approfondimenti, musica, eventi on-line.

Il candidato, formulate le ipotesi aggiuntive e/o semplificative che ritiene necessarie:

- a) proponga e illustri un primo progetto di massima del sistema hardware/software che comporti la installazione del server web presso la redazione;
- b) proponga e illustri un secondo progetto di massima del sistema che comporti l'utilizzo di apparecchiature in hosting presso un provider ISP;
- c) illustri i pro e i contro di ciascuna delle due soluzioni proposte al punto a) e b);
- d) proponga e motivi la soluzione eventualmente mista che a suo parere meglio si adatta alle richieste dell'editore;
- e) illustri il progetto organizzativo necessario al mantenimento del sistema proposto al punto d).

Tema n. 2

Si vuole realizzare un sistema di controllo automatico a microcontrollore o a microprocessore per l'irrigazione del parco di una villa di campagna. Quest'ultimo è suddiviso in quattro zone che debbono essere annaffiate in ore e con tempi diversi secondo la seguente tabella:

Zona	Ora	Tempo in minuti
1	20.00	20'
2	21.00	30'
3	22.00	30'
4	23.00	40'



Ministero dell'Istruzione, dell'Università e della Ricerca

M048 - ESAME DI STATO DI ISTITUTO PROFESSIONALE

CORSO DI ORDINAMENTO

Indirizzo: TECNICO DELLE INDUSTRIE ELETTRONICHE

Tema di: SISTEMI, AUTOMAZIONE E ORGANIZZAZIONE DELLA PRODUZIONE

La durata dei tempi dell'irrigazione è anche legata alla temperatura media T_m della giornata, monitorata ogni ora tra le ore 7.00 e 19.00, come indicato nella tabella sottostante:

$T_m \leq 15^\circ\text{C}$	Il sistema non si avvia
$15^\circ\text{C} < T_m \leq 25^\circ\text{C}$	I tempi sono quelli indicati in tabella
$25^\circ\text{C} < T_m \leq 30^\circ\text{C}$	I tempi debbono essere aumentati del 30%.
$T_m > 30^\circ\text{C}$	I tempi debbono essere aumentati del 50%.

Il sistema di irrigazione, che è azionato da una pompa ad immersione sempre sotto tensione, posizionata sul fondo di un pozzo di 30 m, invia l'acqua alle zone tramite l'apertura e la chiusura programmata di 4 valvole; se il livello dell'acqua si abbassa fino a raggiungere il livello di 2 m rispetto al fondo, il sistema d'irrigazione si deve arrestare. Tale condizione viene segnalata con l'invio di un segnale digitale proveniente da un sensore di livello.

Per acquisire la temperatura si utilizza un sensore che dà in uscita una tensione proporzionale alla temperatura assoluta $V = KT$ con $K = 10 \text{ mV}/^\circ\text{K}$.

Il candidato, formulate le ipotesi aggiuntive che ritiene opportune, e scelto un dispositivo programmabile di sua conoscenza:

1. Descriva tramite schema a blocchi la struttura del controllo.
2. Illustri la funzione dei singoli blocchi.
3. Determini le caratteristiche di ciascun blocco in funzione dei segnali elettrici di ingresso e di uscita.
4. Disegni il flow-chart del programma di gestione.
5. Traduca un segmento del programma in un linguaggio di sua conoscenza.

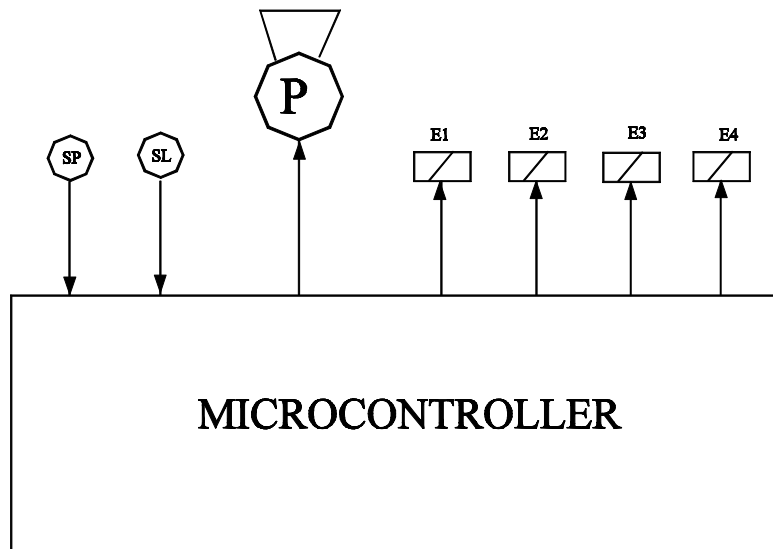
Durata massima della prova: 6 ore.

È consentito soltanto l'uso di manuali tecnici e di calcolatrici tascabili non programmabili.

Non è consentito lasciare l'Istituto prima che siano trascorse 3 ore dalla dettatura del tema.

Soluzione Compito Esame Maturita' Sistemi – Anno 2005/2006

1)



ST: sensore di temperatura + circuito di condizionamento
SL: sensore di livello ON/OFF
P: pompa + azionamento
E1÷E4: elettrovalvole

Il sistema di controllo contiene:

- CPU a 8 bit
- ROM (EPROM) che contiene il programma di gestione e le costanti di sistema
- RAM che contiene le acquisizioni di temperatura e i tempi di irrigazione aggiornabili in funzione delle condizioni ambientali
- Convertitore A/D da 8 bit necessario a convertire il segnale analogico inviato dal sensore di temperatura.
- Interfaccia di uscita di tipo parallelo configurata per acquisire gli ingressi SL ed ST e pilotare, in modo adeguato, le uscite E1÷E4 e P.

2) La funzione svolta dal blocco **ST** è quella di fornire un segnale analogico dipendente dalla temperatura dell'ambiente. Il range di valori forniti dal sensore è:

$$T[^{\circ}K] = \theta[^{\circ}C] + 273$$

$$V = K \cdot T = K \cdot (\theta + 273) = K \cdot \theta + K \cdot 273$$

$$= K \cdot \theta + 2,73 \text{ [volt]} \begin{pmatrix} \theta = 40^{\circ} \rightarrow V = 0,4V \\ \theta = 0^{\circ} \rightarrow V = 2,73V \end{pmatrix}$$

Il blocco **SL**, sensore di livello, ha la funzione di produrre il segnale binario:

- livello basso 0V (0 logico) quando il livello dell'acqua è sopra i 2 m
- livello alto 5V (1 logico) quando il livello dell'acqua è sotto i 2 m

Le quattro elettrovalvole **E1÷E4** sono le uscite del sistema di tipo ON/OFF

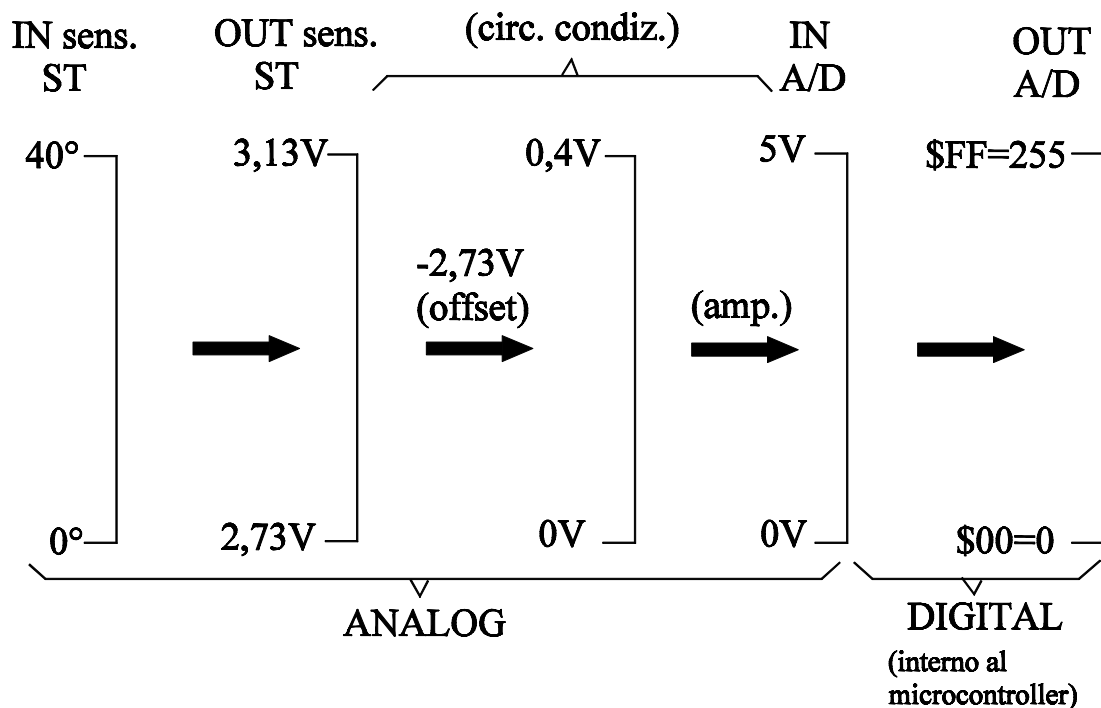
- livello basso 0V (0 logico) chiude l'elettrovalvola
- livello alto 5V (1 logico) apre l'elettrovalvola

Il **microcontroller**, oltre a svolgere la funzione di “orologio”, per determinare gli orari di intervento sulle elettrovalvole e di acquisizione delle temperature, svolge la funzione di controllo che si articola nei tre punti fondamentali:

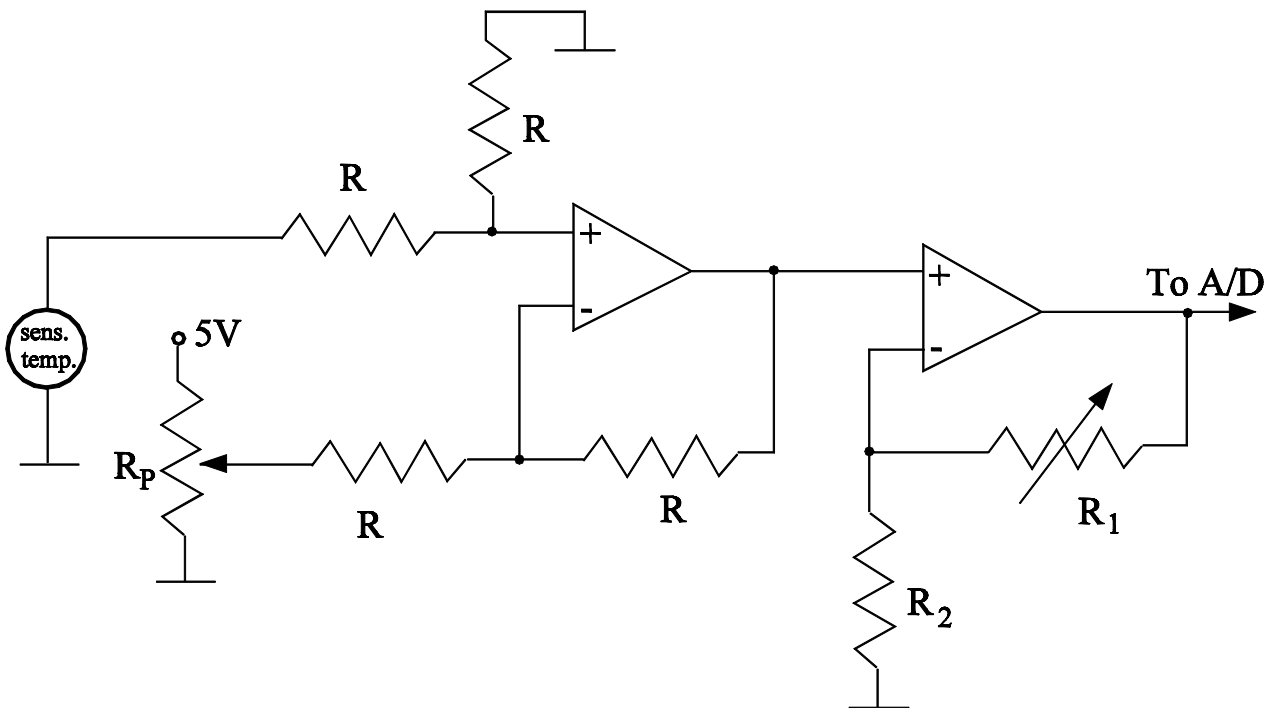
- acquisizione dei segnali dai sensori
- elaborazione di queste informazioni
- attivazione appropriata delle elettrovalvole

Si è inoltre previsto un “comando pompa”, da inviare all'azionamento, nel caso l'acqua del pozzo sia sotto ai due metri. In questo modo, oltre a salvaguardare la pompa, si è pensato di porre il sistema in stand-by (basso consumo), pur mantenendo sotto tensione il sistema controllato (ipotesi aggiuntiva).

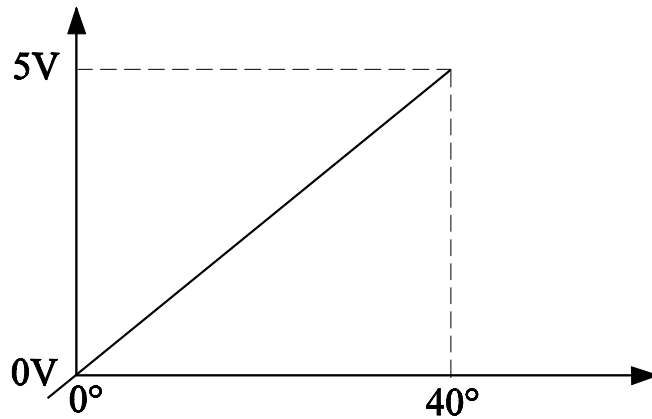
3) Per il blocco **ST**, sensore di temperatura, è prevista una escursione del segnale di temperatura da 0° a 40°:



Dalla relazione ingresso-uscita , $V=KT$, di questo sensore, si rileva la presenza di un **offset** nel segnale di uscita che può essere eliminato con un circuito sottrattore analogico che usa un operazionale. E' necessario, inoltre, un circuito amplificatore capace di garantire una tensione adeguata a pilotare un convertitore A/D, in modo da poter utilizzare tutta la sua scala.

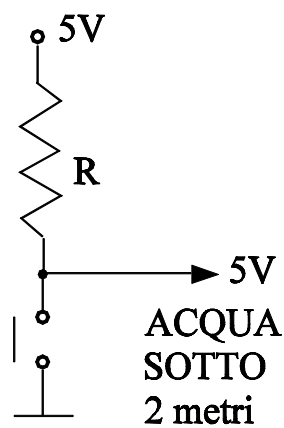
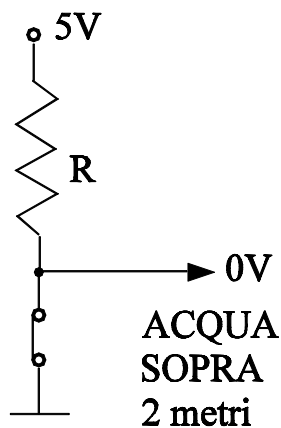


Questi due circuiti realizzano il circuito di condizionamento del segnale che produce, per variazioni di temperatura nel range da 0° a 40° , una corrispondente variazione lineare del segnale di tensione da 0V a 5V (ingresso del convertitore A/D).



Il blocco SL può essere un semplice contatto con galleggiante posizionato a 2 metri di altezza dal fondo del pozzo. I segnali prodotti da questo sensore sono:

- livello basso 0V (0 logico) acqua sopra il livello dei due metri
- livello alto 5V (1 logico) acqua sotto il livello dei due metri



I blocchi E1÷E4 contengono le elettrovalvole e l'elettronica di potenza necessaria a pilotarle

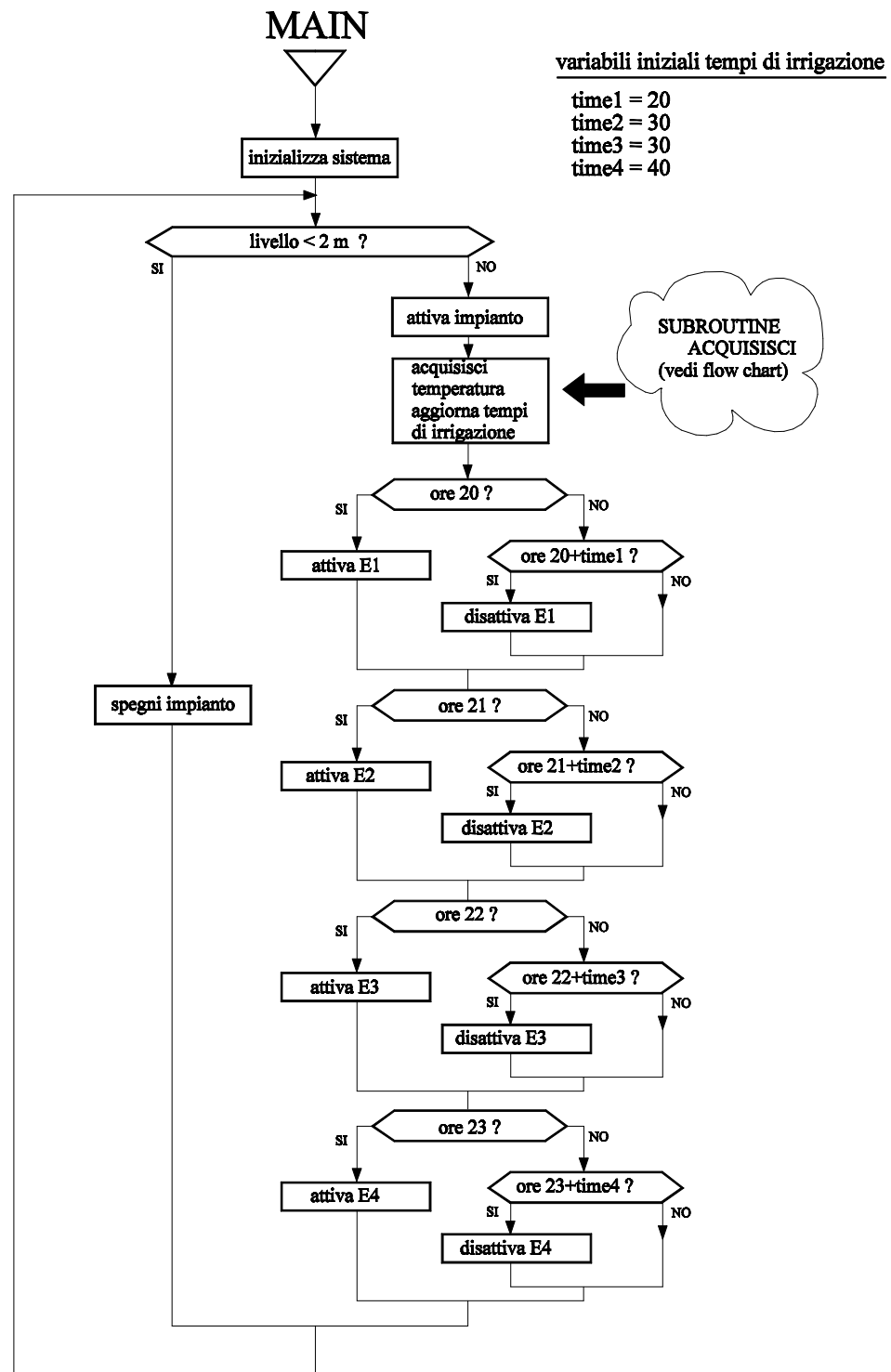
- livello basso 0V (0 logico) inviato alla suddetta elettronica produce la chiusura dell'elettrovalvola
- livello alto 5V (1 logico) apre l'elettrovalvola e consente l'irrigazione

Il blocco P comprende la pompa e l'azionamento di potenza adatto a pilotarla. Si prevede un funzionamento normale, con acqua nel pozzo sopra i due metri, per il quale la pompa è costantemente sotto tensione. Un funzionamento di allarme con acqua sotto ai due metri, che prevede il reset del sistema e posizionamento in stand-by del sistema pompa-azionamento.

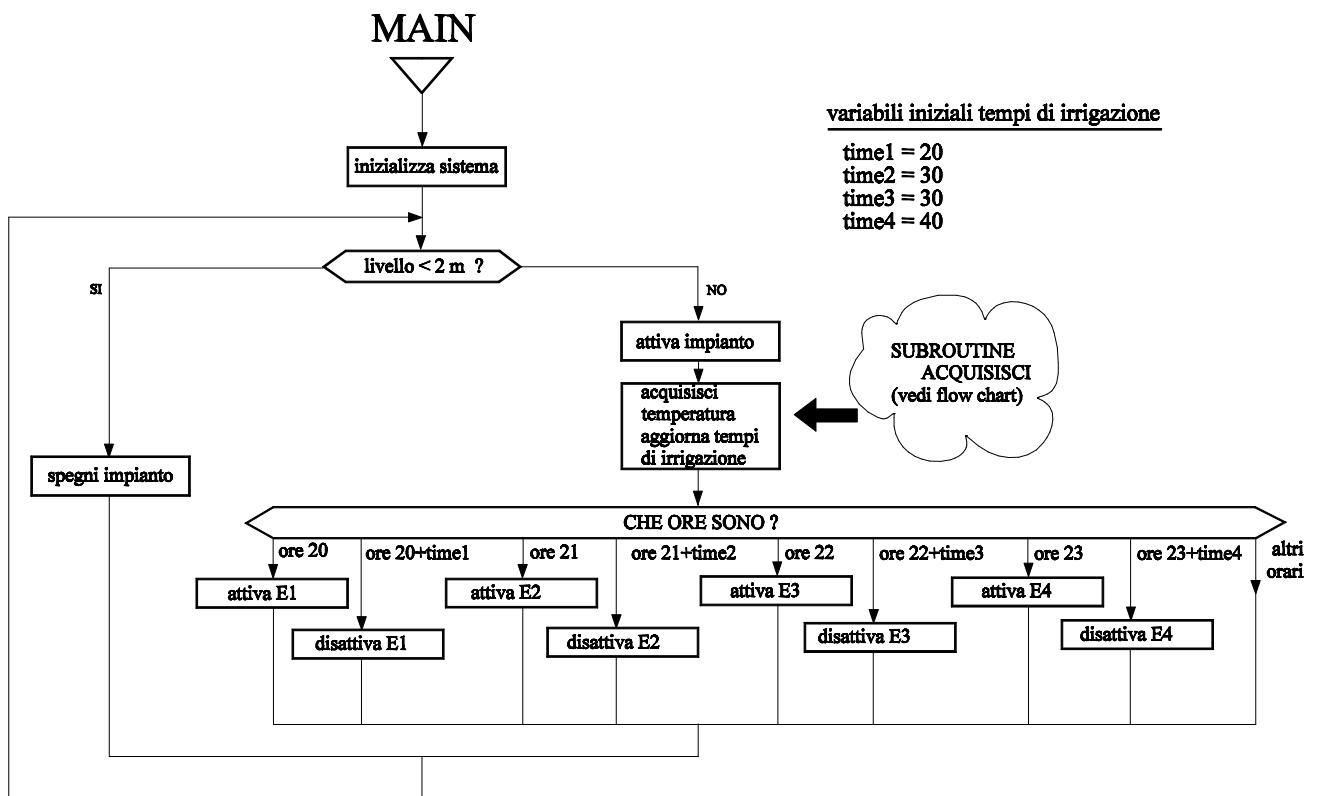
- livello basso 0V (0 logico) pompa disattivata
- livello alto 5V (1 logico) pompa attivata

Infine il microcontroller, in funzione dei segnali di I/O esaminati, non deve avere stringenti caratteristiche di controllo in tempo reale. La sua azione è in catena aperta. I tempi di risposta, senza penalizzare le prestazioni del sistema, possono essere dell'ordine dei secondi o, anche, di alcune decine di secondi. La misura delle temperature può tollerare precisioni anche del 5÷10%, senza che per questo venga compromessa la vegetazione del parco.

4) Flow-chart del programma principale



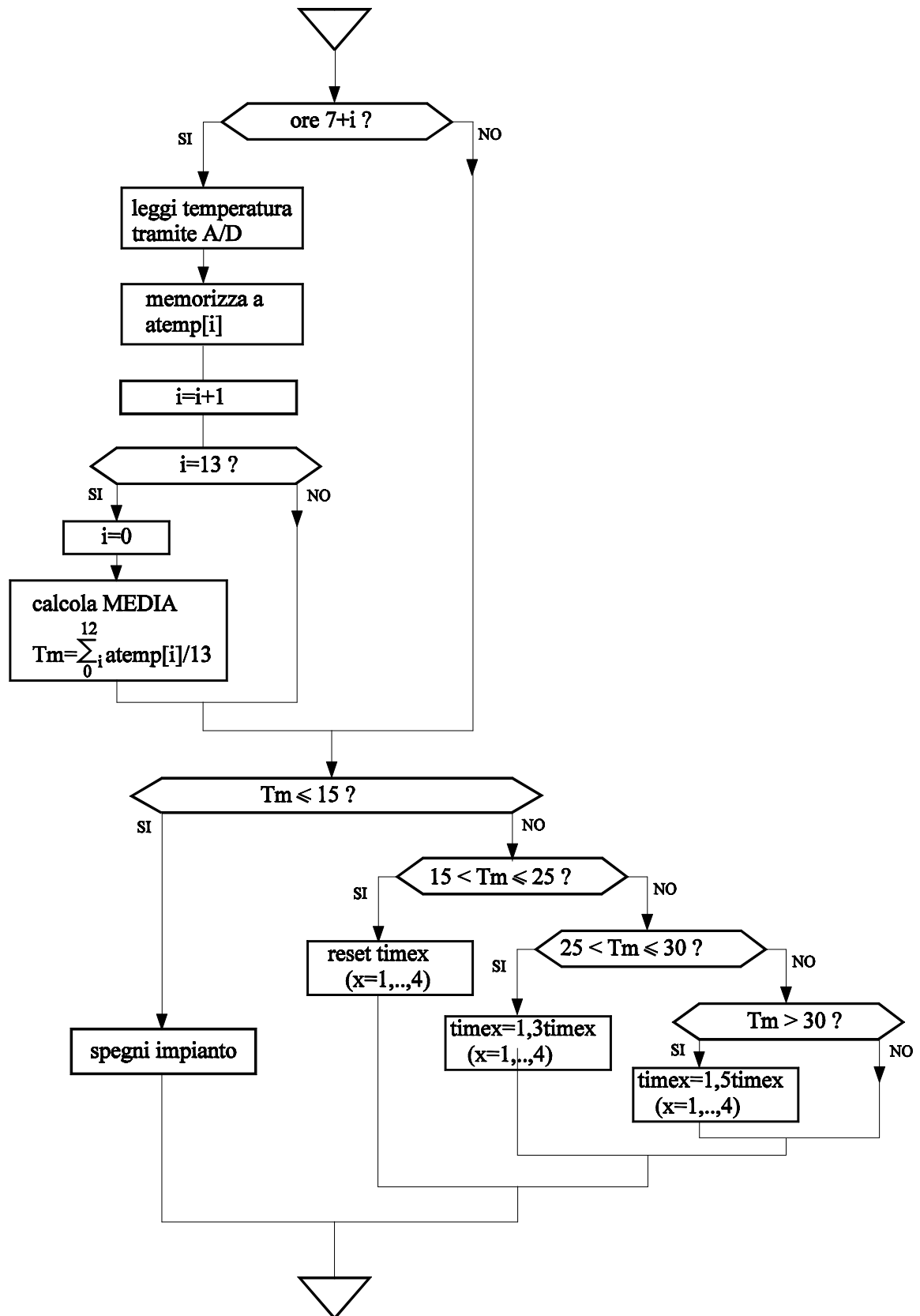
Una versione, logicamente equivalente, dello stesso flow-chart è la seguente:



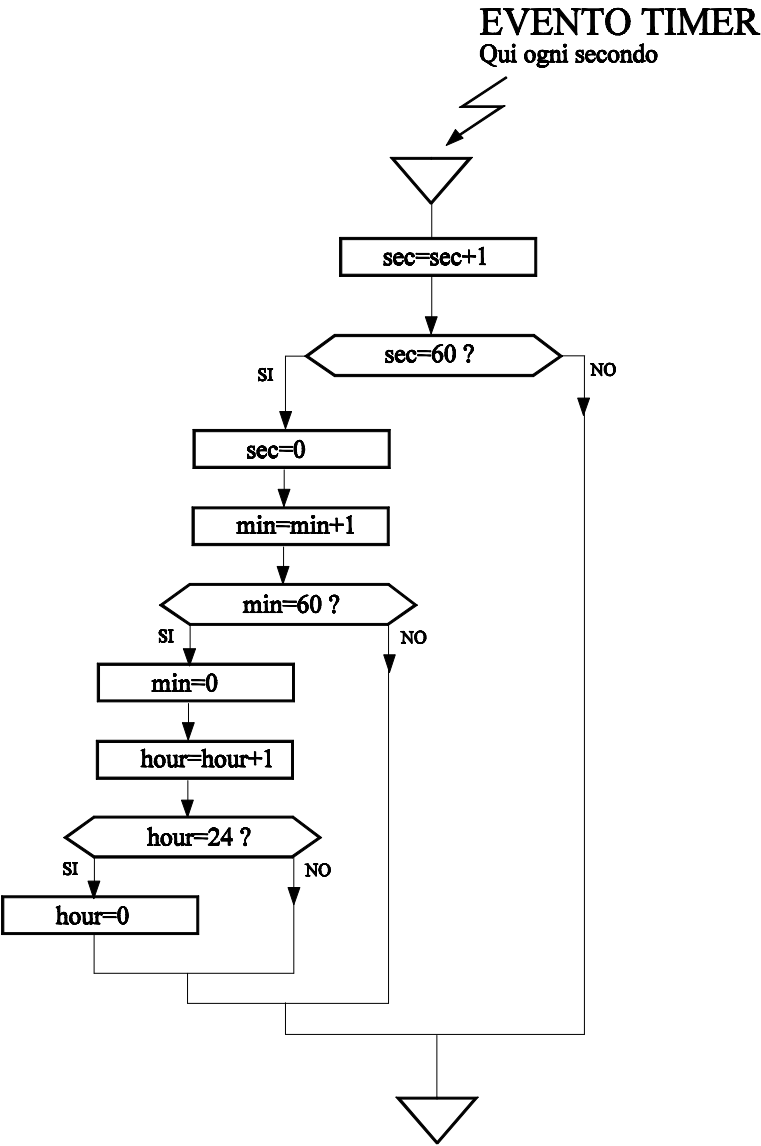
Questa versione suggerisce di implementare questo programma con l'uso della istruzione switch (in C#) mentre la precedente porta a scrivere questa routine con una sequenza di istruzioni if.

Subroutine di acquisizione temperature e calcolo tempi di irrigazione

Subroutine ACQUISISCI



Flow-chart della risposta all'evento timer (interrupt) programmato ogni secondo



5) L'intero programma di irrigazione, del quale il candidato poteva scrivere solo una delle routine che lo compongono, è scritto in C per il processore PIC P18F8520. Per consentire il collaudo con emulatore di questo processore si è ridotta la scala dei tempi. Il giorno solare è stato ridotto ad un giorno fittizio (lo chiameremo "giorno del colibrì") in cui il giorno dura due ore, un ora dura sei minuti un minuto dura sei secondi

```

/*
  Impianto di irrigazione
*/

char time1=2, time2=3, time3=3, time4=4;
char livello = 0;
char ore=0, minuti=0, secondi=0, dcmsec=10;
char step=0;
char atemp[13]; //memoria 13 valori di temperatura da ore 7 a ore 19
char tmedia=0;
char orari[8]; //orari ON/OFF elettrovalvole in minuti
char orario=0; //orario corrente

void interrupt () {
  if(PIR1.TMR1IF) {
    TMR1H=0x85; //10MHz/4=2.5MHz, 2.5MHz/8=312500Hz, T=1/312500=3.2 microsec
    TMR1L=0xEE; //0.1sec/3.2usec=31250 65536-31250=34286=0x85EE
    dcmsec--;
    if(dcmsec == 0) {
      //qui ogni secondo
      PORTC = orario; //oppure PORTC ^ 0x20; //test interrupt
      dcmsec = 10; secondi++;
      if (secondi==6) {
        secondi=0; minuti++;
        if (minuti==6) {
          minuti=0; ore++;
          if (ore==2) ore=0;
        }
      }
    }
  }
  if(orario<2*6*6) {
    orario=ore*6*6+minuti*6+secondi;
  }
  else {
    orario=0;
  }
  PIR1.TMR1IF = 0; //reset flag di interrupt
}

```

```

void initTimer1() {
    INTCON.GIE = 1; // Abilita interrupt globale
    INTCON.PEIE = 1; // Abilita interrupt periferiche
    T1CON.T1CKPS1 = 1; // Prescaler 1:8
    T1CON.T1CKPS0 = 1;
    T1CON.TMR1CS = 0; //Clock interno (Fosc4)
    PIR1.TMR1IF = 0; //Reset bit si stato TIMER1. Posto a 1 da interrupt
    PIE1.TMR1IE = 1; //Abilita interrupt TIMER1
    T1CON.TMR1ON = 1; //Abilita Timer1
    TMR1H=0x85; //10MHz/4=2.5MHz, 2.5MHz/8=312500Hz, T=1/312500=3.2 microsec
    TMR1L=0xEE; //0.1sec/3.2usec=31250 65536-31250=34286=0x85EE
}

```

//Calcola orari accensione e spegnimento elettrovalvole in minuti

```

void neworari() {
    orari[0]=2*60;//20*60;
    orari[1]=3*60+time1;//3*60;//+time1;//20*60+time1;
    orari[2]=4*60;//21*60;
    orari[3]=5*60+time2;//21*60+time2;
    orari[4]=6*60;//22*60;
    orari[5]=7*60+time3;//22*60+time3;
    orari[6]=8*60;//23*60;
    orari[7]=9*60+time4;//23*60+time4;
}

```

//Inizializza convertitore AD

```

void ADCinit() {
    TRISA=0xFF; //bit3-2 input A/D emulatore
    ADCON0=0x08; // bit5-bit2=0010 seleziona input RA2, bit1=0 non conv
    ADCON1=0x00; //bit5-4=00 reference Vdd e Vss, bit3-0=0000 input analogici
    ADCON2=0x85; //bit7=1 res. right allineato, bit2-0=101 Fosc/16
    PIR1.ADIF=0, //reset flag interrupt ADC
    PIE1.ADIE=0; //disabilita interrupt ADC
}

```

//Acquisisci un campione analogico dal canale di ingresso indicato da "ch"

//Risultato con precisione 8 bit. Si rinunciano a due bit di precisione

```

char ADCread(char ch){
    unsigned int ctemp; // contiene lettura corrente temperatura;
    ADCON0=ADCON0&0xC3; //cancella numero canale precedente
    ADCON0=ADCON0|(ch<<2); //inserisci in bit5-2 nuovo numero di canale
    ADCON0.ADON=1; // abilita ADC bit0=1
    Delay_us(50); // aspetta un po' prima dello start conversion
    ADCON0.GO=1; // bit1=1 START conversion
    while(ADCON0.GO); //loop attesa, esci quando bit1=0: fine conversione
    ctemp=ADRESH;
    ctemp=ctemp<<8;
    ctemp=(ctemp|ADRESL)&0x3FF; //valore acquisito 10 bit: max 1023
    return (ctemp*255)/1023;
}

```

```

//Acquisisci temperature ad ogni orario previsto e aggiorna tempi irrigazione
void acquisisci() {
    int i;
    if((ore>=0&&ore<2)&&(minuti==0)&&(secondi==0)) {
        atemp[step]=ADCreed(2);//acquisisci da RA2 temperatura e memorizza
        step++;
    }
    if(step==2) {
        step=0;
        tmedia=0;
        for(i=0;i<2;i++) {
            tmedia=tmedia+atemp[i];
        }
        tmedia=tmedia/2;
    }
    if(tmedia<=15) {
        PORTD=PORTD&0xF0; //spegni elettrovalvole, reset uscite
    }
    else if((tmedia>15)&&(tmedia<=25)) {
        time1=2; time2=3; time3=3; time4=4;
    }
    else if((tmedia>25)&&(tmedia<=30)) {
        time1=1.3*2; time2=1.3*3;
        time3=1.3*3; time4=1.3*4;
    }
    else if(tmedia>=30) {
        time1=1.5*2; time2=1.5*3;
        time3=1.5*3; time4=1.5*4;
    }
    neworari();//aggiorna orari intervalli di irrigazione misurati in minuti
}

```

```

void main() {

    PORTD = 0x00; // E1,E2,E3,E4: 0; elettrovalvole chiuse
    TRISD = 0x30; //da bit0 a bit3: E1,E2,E3,E4, bit4: SL, bit5: ST, bit6:pompa
    PORTC = 0x00;
    TRISC = 0x00; //test interrupt
    ADCinit();    //inizializza convertitore ADC
    initTimer1();
    neworari();//aggiorna orari intervalli di irrigazione misurati in secondi
    while(1){
        livello=PORTD&0x010;
        if(livello==0) {
            PORTD=PORTD|0x40; //bit 6=1 azionamento-pompa attivo
            acquisisci();
            if(orario==orari[0]) {
                PORTD=PORTD|0x01; //attiva E1
            }
            else if (orario==orari[1]) {
                PORTD=PORTD&0xFE; //disattiva E1
            }
            else if (orario==orari[2]) {
                PORTD=PORTD|0x02; //attiva E2
            }
            else if (orario==orari[3]) {
                PORTD=PORTD&0xFD; //disattiva E2
            }
            else if (orario==orari[4]) {
                PORTD=PORTD|0x04; //attiva E3
            }
            else if (orario==orari[5]) {
                PORTD=PORTD&0xFB; //disattiva E3
            }
            else if (orario==orari[6]) {
                PORTD=PORTD|0x08; //attiva E4
            }
            else if (orario==orari[7]) {
                PORTD=PORTD&0xF7; //disattiva E4
            }
        }
        else {
            PORTD=PORTD&0x00; //reset bit 6 azionamento-pompa standby
        }
    }
}

```