

**Istituto Professionale di Stato per l'Industria e l'Artigianato
MORETTO
Via Luigi Apollonio, 21 BRESCIA**

Controllo e monitoraggio temperatura mediante PC

Realizzato da :

RIZZINELLI ALESSANDRO

VARISCO ANGELO

COMINI PAOLO

classe 5BI TIEE

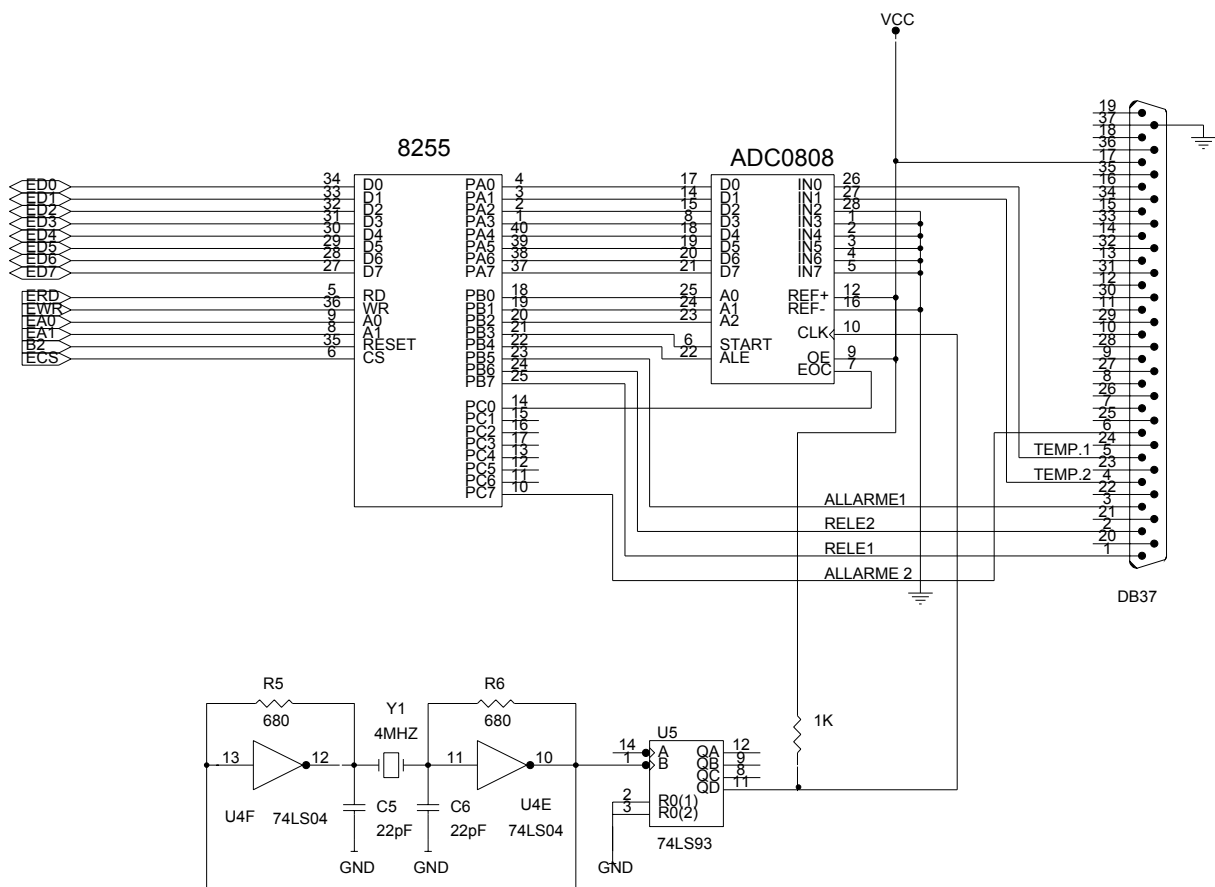
corso per Tecnici delle Industrie Elettriche ed Elettroniche

anno scolastico 1996-97

PREMESSA SUL MONITORAGGIO E CONTROLLO TERMICO	3
CARATTERISTICHE GENERALI DI UN ADC : CONVERTITORE ANALOGICO-DIGITALE.....	3
<i>Precisione</i>	4
<i>Sensibilità</i>	4
<i>Potere risolutivo</i>	4
<i>Tempo di conversione</i>	4
IL CONVERTITORE ADC0808/0809	4
INTERFACCIA PERIFERICA PROGRAMMABILE 8255.....	6
<i>DATA BUS BUFFER</i>	7
<i>READ / WRITE and CONTROL LOGIC</i>	7
MODALITÀ DI FUNZIONAMENTO I/O	8
<i>HANDSHAKE o "stretta di mano"</i>	9
INTERFACCIA SPERIMENTALE LX833	11
<i>Segnali di controllo provenienti dal Bus del PC</i>	11
<i>Segnali di controllo presenti sulla scheda</i>	12
<i>Altri segnali presenti sulla scheda</i>	12
<i>Oscillatore a Quarzo 500 KHz</i>	12
<i>Funzionamento</i>	12
<i>Funzionamento Cristallo Di Quarzo</i>	13
DESCRIZIONE DELLA PROGRAMMAZIONE IN PASCAL.....	13
IMPOSTAZIONE GENERALE DELL'ELABORATO PER IL CONTROLLO DI TEMPERATURA.	14
PROGRAMMA SORGENTE REDATTO IN TURBO PASCAL 6.0.....	15

Premessa sul monitoraggio e controllo termico

Lo scopo della nostra prova di laboratorio è quello di misurare le temperature di due vasche in cui si sviluppa un processo produttivo industriale. Le due temperature devono essere monitorate sullo schermo di un PC. Si deve perciò realizzare un apparato che agisca da interfaccia fra i trasduttori di temperatura con scala 0-5V con temperatura compresa fra 0 e 300 °C e il PC (Amstrad PC1640). Per realizzare questo abbiamo utilizzato una scheda di interfaccia parallela LX 833 prodotta da “Nuova Elettronica” su cui abbiamo montato un 82C55A : adattatore di interfaccia parallelo programmabile della famiglia INTEL e un convertitore analogico/digitale ADC0808 della National.



Caratteristiche generali di un ADC : Convertitore Analogico-Digitale

La funzione dei convertitori A/D è quella di trasformare un livello di tensione in un numero N espresso in codice binario ad esso corrispondente come esplicitato dalla formula:

$$N = \frac{V_I}{V_{REF}} \cdot 2^n$$

dove $\frac{V_{REF}}{2^n}$ è la risoluzione del convertitore e n è il numero dei bit del convertitore A/D.

Precisione

E' definita dallo scarto tra il valore convertito ed il valore vero della grandezza in esame. Le cause che alterano la precisione del convertitore sono essenzialmente :

- la deriva, intesa come effetto della temperatura, dell'invecchiamento dei componenti e delle fluttuazioni dell' alimentazione;
- la non linearità del comportamento.

Sensibilità

E' rappresentata dal più piccolo livello di tensione che può essere convertito. Si definisce anche il massimo valore dei segnali di ingresso che consentono al convertitore di lavorare nelle migliori condizioni di funzionamento.

Potere risolutivo

E' la più piccola variazione di tensione cui è sensibile il convertitore, questo non è da confondere con la sensibilità in quanto il convertitore può non avvertire tensioni prossime allo zero ma può sentire piccole variazioni in un punto qualsiasi delle sue caratteristiche di comportamento dinamico.

Tempo di conversione

Rappresenta il tempo impiegato dal circuito per eseguire la conversione da analogico in digitale. Gli elementi che lo compongono sono:

- tempo di conversione incrementale: è il tempo impiegato a convertire una tensione corrispondente al potere risolutivo, cioè al minimo incremento cui il circuito è sensibile;
- tempo di conversione sull'intera scala: tempo impiegato dal circuito ad effettuare la conversione della massima tensione misurabile (a partire dal livello 0).

Il Convertitore ADC0808/0809

Il circuito integrato ADC0808 è un dispositivo CMOS monolitico convertitore A/D ad 8 bit corredato di multiplexer a 8 canali e di logica di controllo compatibile con sistemi a microprocessore. Il circuito utilizza la tecnica delle approssimazioni successive nel sistema di conversione ; presenta inoltre un comparatore stabilizzato “chopper” ad alta impedenza, un partitore di tensione a 256 resistenze con un albero di interruttori analogici e un registro SAR . Il multiplexer ad 8 canali può accedere direttamente a uno degli 8 segnali analogici con massa in comune. Il dispositivo elimina il bisogno di una massa esterna e di una regolazione di fondo scala.

Il componente offre alta velocità, alta precisione, minima dipendenza dalla temperatura, e consuma una minima energia.

L' ADC 0808 presenta le seguenti caratteristiche :

- risoluzione a 8 bit ;
- errore totale non correggibile compreso fra +/- 1/2 LSB e +/- 1 LSB ;
- alimentazione singola pari a 5V ;
- potenza dissipata 15 mW ;
- tempo di conversione pari a 100 us ;

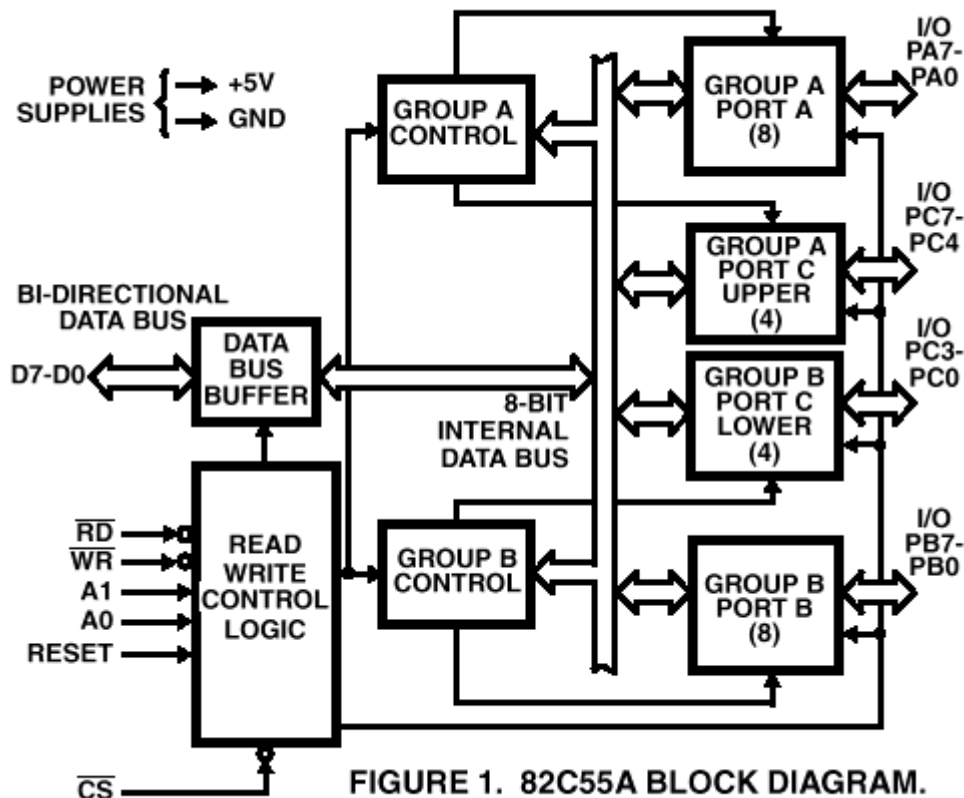
IL CONVERTITORE A/D

Il registro ad approssimazioni successive (SAR) esegue 8 iterazioni per effettuare la misura della tensione d'ingresso. Il SAR del convertitore A/D si resetta sul fronte di salita del segnale di inizio conversione SC. La conversione ha inizio sul fronte di discesa del segnale di inizio conversione SC (modalità di funzionamento monostabile). Una conversione in atto verrà interrotta da un nuovo segnale di inizio conversione SC. Si possono ottenere conversioni continue collegando l' uscita di fine conversione (EOC) all' ingresso SC (modalità di funzionamento astabile). Se usato in questo modo, è necessario fornire un impulso esterno di inizio conversione subito dopo avere dato tensione al convertitore. L'uscita EOC (fine conversione) andrà a livello basso in un tempo compreso fra 1 e 8 periodi di clock a partire dal fronte di salita del segnale SC. La più importante sezione del convertitore A/D è il comparatore. Questa sezione è responsabile della precisione finale dell' intero convertitore. Anche la deriva del comparatore ha una grande influenza sulla ripetibilità del dispositivo. Un comparatore stabilizzato di tipo “chopper” fornisce il metodo più efficace per soddisfare tutte le richieste del convertitore. Il comparatore stabilizzato chopper converte il segnale continuo d'ingresso in un segnale alternato. Questo segnale viene introdotto in un amplificatore in alternata ad elevato guadagno, in uscita viene ripristinato il livello continuo. Questa metodo limita considerevolmente gli effetti negativi della deriva dei componenti dell'amplificatore dato che la deriva di un componente in continua non passa attraverso un amplificatore in corrente alternata. Questo fa sì che l'intero convertitore sia estremamente insensibile alla temperatura, alla deriva per lunghi periodi e agli errori di offset.

	C	B	A
IN0	L	L	L
IN1	L	L	H
IN2	L	H	L
IN3	L	H	H
IN4	H	L	L
IN5	H	L	H
IN6	H	H	L
IN7	H	H	H

Interfaccia periferica programmabile 8255

L' 8255A è un dispositivo di interfacciamento periferico programmabile, fabbricato per essere usato con microprocessori della famiglia INTEL. La sua funzione è quella di consentire la realizzazione di canali di I/O di tipo generale, ciò servirà per interfacciare un generico dispositivo periferico esterno al bus del microcomputer. La configurazione dell'8255A avviene attraverso un programma (software di inizializzazione) che può essere redatto in linguaggio assembler o in linguaggi ad alto livello (Pascal, Basic, C). Poiché l'8255 è un dispositivo programmabile, normalmente non è necessario corredare il circuito di particolari accorgimenti circuitali (hardware) esterni per interfacciare dispositivi periferici esterni anche complessi. Nella figura viene riportato uno schema funzionale dell'8255 ove sono messi in evidenza i due ambienti: il bus del microprocessore (Data Bus, Address Bus e Control Bus) e l'ambiente esterno ove sarà collocata l'applicazione specifica connessa all'8255 attraverso le linee periferiche del Port A, del Port B e del Port C. Nella figura che segue è riportato uno schema a blocchi più dettagliato del dispositivo di interfaccia da noi usato.



Commentiamo ora i blocchi principali :

DATA BUS BUFFER

E' di tipo bidirezionale in tecnologia "tri-state" ed é usato per interfacciare l' 8255 al data bus del sistema. I dati sono trasmessi o ricevuti dal buffer, durante l'esecuzione di istruzioni di ingresso o uscita da parte della CPU. I segnali di controllo e le informazioni di stato della periferica vengono solo trasferite attraverso il data buffer.

READ / WRITE and CONTROL LOGIC

La funzione di questo blocco é di manovrare tutti i trasferimenti interni ed esterni di dati informazioni di controllo o di stato della periferica. Esso è connesso attraverso ingressi all'Address Bus e al Control Bus e fornisce segnali in Uscita sul Control Bus.

(CS) selezione circuito

Un livello basso "low" su questo pin di ingresso mette in grado di comunicare l' 8255 con la CPU.

(RD) leggere

Un livello "low" su questo pin di ingresso mette in grado l' 8255 di trasmettere i dati e lo stato delle informazioni al CPU tramite i data bus. In sostanza questa funzione permette alla CPU di leggere dall'8255.

(WR) scrivere

Un livello basso "low" su questo pin di ingresso mette in grado la CPU di scrivere i dati o informazioni di controllo dentro l'8255.

(A0 and A1) ingressi di indirizzamento

Questi ingressi, unitamente allo stato delle linee WR e RD, controllano la selezione di uno dei 3 ports o dei registri di controllo interni dell'8255. Essi vengono normalmente connessi ai bit meno significativi dell'address bus AB0 e AB1.

(RESET) cancellare

Un livello alto "high" su questo ingresso porta a zero il contenuto del registro di controllo ; i Port A,B,C vengono posti in modalità Input con circuiteria "Bus Hold" attivata.

82C55A BASIC OPERATION

A1	A0	RD	WR	CS	INPUT OPERATION (READ)
0	0	0	1	0	Port A → Data Bus
0	1	0	1	0	Port B → Data Bus
1	0	0	1	0	Port C → Data Bus
1	1	0	1	0	Control Word → Data Bus
OUTPUT OPERATION (WRITE)					
0	0	1	0	0	Data Bus → Port A
0	1	1	0	0	Data Bus → Port B
1	0	1	0	0	Data Bus → Port C
1	1	1	0	0	Data Bus → Control
DISABLE FUNCTION					
X	X	X	X	1	Data Bus → Three-State
X	X	1	1	0	Data Bus → Three-State

Modalità di funzionamento I/O

L' INTEL 8255A é un dispositivo di I/O (interfaccia programmabile) progettato per essere usato con la famiglia di microprocessori INTEL. Esso ha 24 pin di I/O che possono essere individualmente programmati in due gruppi da 12 e usati in tre principali modalità di funzionamento.

Nella “modalità 0” (MODE 0) sono possibili 16 differenti configurazioni di I/O come mostrato in tabella. Questo modo di funzionare consente di effettuare semplici operazioni di Input e Output attraverso i tre port. Il Port A (da PA0 a PA7) ed il Port B (da PB0 a PB7) possono essere programmati in modo Input o Output nel Port C si possono dividere gli 8 bit in due gruppi : Port C High (da PC4 a PC7) e Port C Low (da PC0 a PC3) ciascun dei quali può essere programmato in modo Input o in modo Output.

In estrema sintesi la modalità 0 consente :

- 1) due port da 8 bit (Port A e Port B) e due port da 4 bit (PCL e PCH) ;
- 2) ciascun port può funzionare da Input o da Output ;
- 3) i port che funzionano da Output sono internamente memorizzati
- 4) i port di ingresso non sono memorizzati
- 5) sono possibili 16 diverse modalità di funzionamento.

Nella figura a fianco è riportata la configurazione da noi prescelta per l'applicazione realizzata.

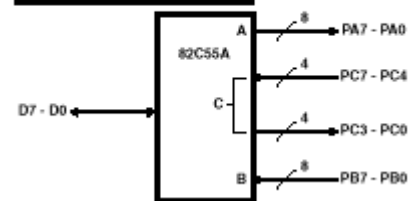
Nella “modalità 1” (MODE 1 o Strobed I/O) il circuito integrato consente di effettuare operazioni di I/O in tecnica “handshake” attraverso i due port (A e B) ; le linee del Port C vengono utilizzate per generare o ricevere i segnali di “handshake”.

MODE 0 PORT DEFINITION

A		B		GROUP A		#	GROUP B	
D4	D3	D1	D0	PORT A	PORTC (Upper)		PORT B	PORTC (Lower)
0	0	0	0	Output	Output	0	Output	Output
0	0	0	1	Output	Output	1	Output	Input
0	0	1	0	Output	Output	2	Input	Output
0	0	1	1	Output	Output	3	Input	Input
0	1	0	0	Output	Input	4	Output	Output
0	1	0	1	Output	Input	5	Output	Input
0	1	1	0	Output	Input	6	Input	Output
0	1	1	1	Output	Input	7	Input	Input
1	0	0	0	Input	Output	8	Output	Output
1	0	0	1	Input	Output	9	Output	Input
1	0	1	0	Input	Output	10	Input	Output
1	0	1	1	Input	Output	11	Input	Input
1	1	0	0	Input	Input	12	Output	Output
1	1	0	1	Input	Input	13	Output	Input
1	1	1	0	Input	Input	14	Input	Output
1	1	1	1	Input	Input	15	Input	Input

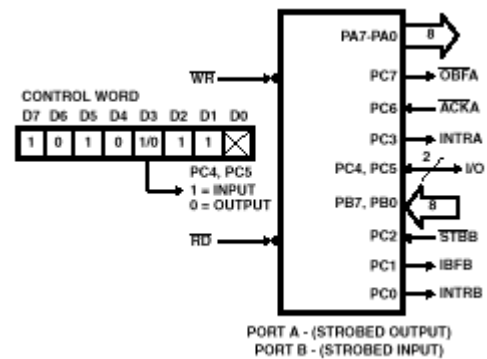
CONTROL WORD #6

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	1	0



Nella figura a fianco è riportato, un esempio di impiego dell'8255 in cui il Port A funziona da uscita, il Port B da ingresso le linee PC7 (Strobe) e PC6 (Acknowledge) realizzano l'hanshake per il Port A, le linee PC1 (STB) e PC2 (ACK) l'hanshake per il Port B.

Sono disponibili pure uscite di Interrupt : PC3 (INTRA) e PC0 (INTRB) attraverso le quali si può pilotare un apposita logica di generazione di richieste di "interrupt da inoltrare alla CPU.



Nella "modalità 2" (MODE 2 o Strobed bidirectional Bus I/O) il circuito integrato consente di effettuare operazioni di I/O bidirezionale sotto il controllo dei segnali di "handshake" in modalità simile al Modo 1 .

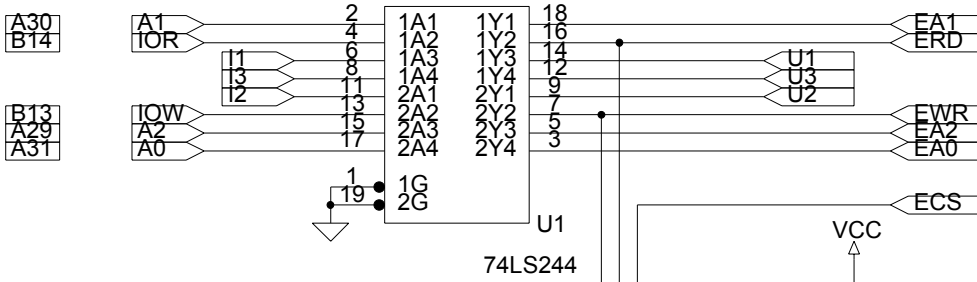
HANDSHAKE o "stretta di mano"

Quando due dispositivi asincroni devono trasferire tra loro informazioni attraverso una porta di comunicazione (di tipo parallelo 4 o 8 bit), per sincronizzare il trasferimento dati utilizzano il metodo handshake o della "stretta di mano". Il dispositivo che trasmette il dato lo colloca sulla porta di comunicazione ; poi fornisce su una linea di controllo ausiliaria, un segnale cosiddetto di "strobe" o di "conferma dato" (in genere un impulso attivo a livello logico basso della durata approssimativa di 1 µs). La linea di controllo distinta dalle linee dati e denominata linea di "strobe" è uscita nel circuito trasmettitore e ingresso nel ricevitore. Il ricevitore avvertito della presenza del dato sulla porta lo legge, lo trasferisce "al sicuro" e ad operazione ultimata conferma con una segnalazione di Data Acknowledge che il dato è stato correttamente acquisito. La segnalazione Data Acknowledge viene inoltrata dal circuito ricevitore che la genera, al circuito trasmettitore che la riceve attraverso una apposita linea di controllo indipendente dalle linee dati. Il trasmettitore può quindi mettere così a disposizione del ricevitore altri dati che vengono sempre trasferiti al ricevitore con questa tecnica della doppia conferma denominata handshake.

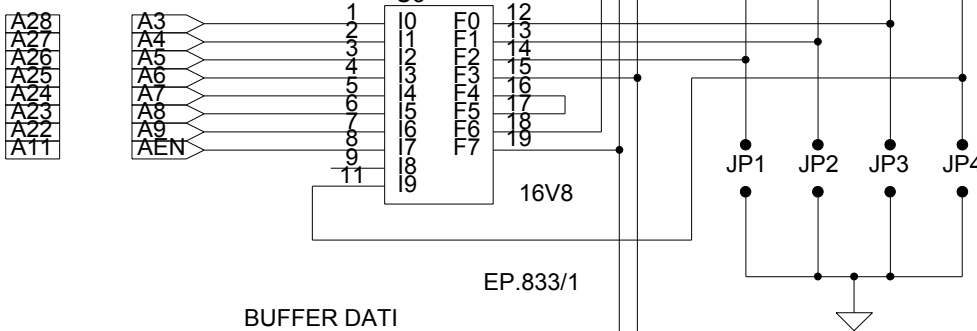
Un classico esempio di "handshake" si incontra nello studio del modo di funzionamento della interfaccia Centronics che sta alla base delle stampanti connesse ai PC con interfacciamento parallelo. Il trasmettitore è il PC o meglio l'interfaccia parallela del PC, il ricevitore è la stampante.

S P C
T

LOGICA DI CONTROLLO



LOGICA DI SELEZIONE

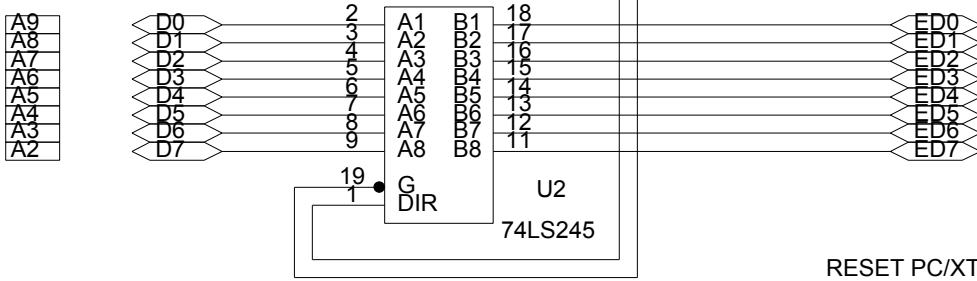


CAMPO DI SELEZIONE

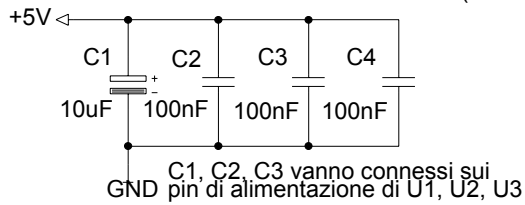
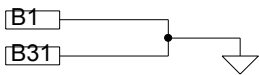
- JP1-ON \$300-\$307
- JP2-ON \$308-\$30F
- JP3-ON \$310-\$317
- JP4-ON \$318-\$31F

BUFFER DATI

BIDIREZIONALE



RESET PC/XT



Interfaccia sperimentale LX833

In fig.1 è riportato lo schema elettrico dell'interfaccia sperimentale LX833 realizzata da Nuova Elettronica e adatta per essere inserita negli slot a 8 bit di un Personal Computer IBM compatibile classe XT oppure AT.

Il circuito LX833 utilizza tre circuiti integrati cui sono affidati i seguenti compiti:

- U1 - 74LS244 *buffer per amplificazione di indirizzi e segnali di controllo*
- U3 - 74LS245 *buffer bidirezionale per dati*
- U2 - PAL 16V8 (*programmable array logic*) *decodifica indirizzi*

Il circuito preleva direttamente dal BUS del PC i segnali presenti sul Data-Bus (D0-D7) sull'Address-Bus (A0-A9) i segnali di controllo: IOR, IOW, AEN e le tensioni di alimentazione (+5V, +12V, -12V) necessarie per la alimentazione sia dei circuiti integrati già presenti nel circuito, sia dei circuiti integrati da collocare in area wrap (area a disposizione dell'utente). La scheda è provvista anche da una serie di jumper (ponticelli) attraverso i quali si stabilisce l'area di indirizzamento più opportuna della scheda LX833.

Un solo jumper alla volta può essere chiuso. Poichè in ingresso alla PAL (U2) giungono tutti gli indirizzi meno i tre meno significativi (A0, A1, A2), ne deriva che l'abilitazione si attiverà su 8 indirizzi di I/O contigui come riportato in tab. 1.

Jumper	CAMPO
JP1	\$300-\$307
JP2	\$308-\$30F
JP3	\$310-\$317
JP4	\$318-\$31F

Segnali di controllo provenienti dal Bus del PC

AO-A1-A2: rappresentano i primi tre bit del bus degli indirizzi.

IOR (Input Output Read): IOR, significa "lettura da un dispositivo di ingresso/uscita". Quando la CPU presente nel PC intende effettuare una operazione READ da un dispositivo di I/O pone a livello logico basso il segnale IOR.

IOW (Input Output Write): IOW, significa "scrittura verso un dispositivo di ingresso/uscita". Quando la CPU presente nel PC intende effettuare una operazione WRITE verso un dispositivo di I/O pone a livello logico basso il segnale IOW.

AEN (Address Enable): AEN significa "abilitazione all'Address Bus". Quando la CPU presente nel PC emette un livello logico basso su AEN ciò significa che l'indirizzo esistente in quell'istante sull'Address Bus è da considerarsi valido.

Segnali di controllo presenti sulla scheda

ECS (Expansion chip select): ECS significa "selezione della scheda di espansione". Lo scopo di questo segnale è di fornire l'abilitazione agli elementi circuitali presenti sulla scheda, quando sul bus degli indirizzi è presente uno degli indirizzi contenuto entro il gruppo selezionato dal jumper chiuso (vedi tab. 1). Il segnale ECS è attivo a livello logico basso.

Altri segnali presenti sulla scheda

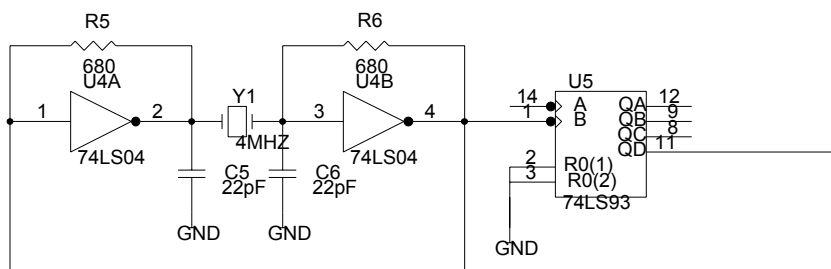
ED0-ED7 rappresentano il Data-Bus bufferizzato D0-D7 a valle di U3 SN74LS245 (transceiver o buffer bidirezionale).

EA0-EA2 versione bufferizzata degli indirizzi bassi A0-A3 a valle di U1 SN74LS244

ERD versione bufferizzata del segnale IOR a valle di U1 SN74LS244

EWR versione bufferizzata del segnale IOW a valle di U1 SN74LS244

Oscillatore a Quarzo 500 KHz



Oscillatore al quarzo con porte not normali (non triggerate), serve per avere una frequenza stabile, questo è formato da diversi elementi fondamentali: due porte not o, due resistenze, due condensatori e un cristallo di quarzo.

Funzionamento

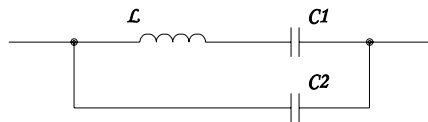
Le porte TTL servono da amplificatori invertenti. Per far questo devono funzionare in zona lineare (non in saturazione né in interdizione) per ottenere questa condizione di funzionamento dobbiamo fare in modo che la tensione di ingresso V_I risulti essere uguale a quella di uscita V_U .

Per ottenere ciò dobbiamo inserire in parallelo alle porte not (cioè tra ingresso e uscita) una resistenza di valore non molto elevato ($R < 1K$), queste due porte not funzionano da oscillatore infatti, funzionando da amplificatori invertenti il segnale che ne uscirà sarà sinusoidale.

Funzionamento Cristallo Di Quarzo

Inserendo il cristallo di quarzo tra le due porte esso funziona da filtro passa banda infatti soltanto quando la reattanza interna sarà circa uguale a zero condurrà e consentirà il passaggio del segnale che avrà una frequenza uguale alla frequenza di serie f_s che che sarà a sua volta uguale circa a quella di parallelo f_p .

Il circuito equivalente al cristallo di quarzo si può sintetizzare come :



Grazie alle porte not e al quarzo si è realizzato un oscillatore preciso di frequenza il cui valore di frequenza dipende dal cristallo utilizzato .

Dato che noi abbiamo utilizzato un cristallo di valore commerciale 4MHz e il nostro integrato ADC 0808 richiede un clock di 500 KHz si metterà in cascata sull' uscita dell'oscillatore un divisore di frequenza per 8.

Descrizione della Programmazione in Pascal

La programmazione in Pascal è un linguaggio adatto a risolvere sia problemi di tipo scientifico che problemi di altro genere ed è strutturato per definizione.

In questo linguaggio ci sono diversi tipi di VARIABILI.

- 1) Tipi semplici
- 2) Tipi strutturati
- 3) Tipo stringa
- 4) Tipo puntatore
- 5) Tipo identificatore

LE VARIABILI piu' importanti sono i tipi semplici ,i tipi strutturati e il quelli stringa.

Quelle semplici possono essere di 2 tipi:

Tipi ordinali(short int,integer,longint,byte,word,boolean,char)e tipi reali.

Gli interi di tipo 'integer' sono molto utili ,ma hanno un utilizzo solo nel campo intero ; per rappresentare numeri con la virgola occorre utilizzare variabili di tipo 'real' i quali danno dei problemi legati soprattutto alla velocita' d'esecuzione ,poiche' la CPU opera molto piu' velocemente sulle variabili intere in quanto richiedono solo 2BYTES.

Nell'utilizzo delle integer si può incorrere in una situazione di 'overflow'che accade quando il risultato di una elaborazione condotta su due quantita' intere produce un valore troppo grande per

poter essere memorizzato nell'area riservata al risultato. In questi casi bisogna utilizzare le variabili di tipo 'Longint' piu' lunghe perché di 4BYTES.

LE STRUTTURE DECISIONALI :serve per determinare fra 2 opzioni quale scegliere.

Questa struttura è costituita da IF, THEN' ,ed infine 'ELSE' con il significato di altrimenti

Un'esempio di procedura decisionale puo' essere cosi' scritto :

```
IF <ESPRESSIONE BOOLEANA>THEN
    <ISTRUZIONE1>
    ELSE <ISTRUZIONE2>
```

IL PROGRAMMA PASCAL è formato da parti fondamentali che devono essere elencate in un ordine prestabilito:

- PROGRAM(scritto all'inizio del programma)
- DEFINIRE LE COSTANTI
- DEFINIRE LE VARIABILI
- DEFINIRE LE FUNZIONI
- PROGRAMMA PRINCIPALE

Impostazione generale dell'elaborato per il controllo di Temperatura.

Si vuole realizzare un programma in pascal che contolli la temperatura di 2 vasche ,per far questo abbiamo utilizzato gli integrati sopra descritti; e dei trasduttori di temperatura per la misurazione delle temperature.

Con il nostro elaborato si puo' impostare una temperatura per ogni vasca (temperatura impostata 1,2) e scegliere una banda percentuale intorno ad essa ,in cui, la temperatura puo essere accettata; in questa banda i rele' per il riscaldamento o raffreddamento dell' acqua sono inattivi. Tutto questo è realizzato grazie al nostro programma in pascal.Inoltre quando le temperature superano un certo valore, interviene un allarme.

Programma sorgente redatto in Turbo Pascal 6.0

```
program ADC;
uses Dos,Crt;
{ Un convertitore ADC mod ADC0808 National Š interfacciato al PC
mediante una periferica parallela 8255 montata su una scheda
di interfaccia LX833 (Nuova Elettronica).
Il PORT-A dell'8255 funziona da INPUT e preleva i dati provenienti
dalle 8 uscite D0 .. D7 del convertitore ADC 808;
Il PORT-B dell'8255 funziona da OUTPUT come segue:
PB0, PB1, PB2 sono collegati sugli ingressi A0, A1, A2
dell'ADC (indirizzamento del canale);
PB3 Š collegato all'ingresso START dell'ADC (inizio conversione)
PB4 Š collegato all'ingresso ALE dell'ADC (memorizza A0,A1,A2)
PB5 Š collegato all'ALLARME1 esterno
PB6 Š collegato al RELE2 esterno
PB7 Š collegato al RELE1 esterno
Il PORT-C dell'8255 per met... da ingresso e per met... da uscita:
PC0 .. PC3 funzionano da ingresso
PC4 .. PC7 funzionano da uscite

PC0 Š collegato all'uscita EOC dell'ADC (fine conversione)
PC7 Š collegato all'ALLARME2 esterno }

const
scheda=$300;
porta=scheda+$0; { solo input }
portb=scheda+$1; { solo output }
porte=scheda+$2; { PC0-PC3=input PC4-PC7=output }
control=scheda+$3; { registro di controllo solo write }
days : array [0..6] of String[9]=('Domenica','Lunedł ', 'Martedł ',
'Mercoledł ', 'Giovedł ', 'Venerdł ',
'Sabato');

Var i: integer;
Mt1, Mt2, TEMP1, T1min, T1max, TEMP2, T2min, T2max, P1, P2,
Tal1, Tal2 : real ;
allarme1, Allarme2 :boolean;
y, m, d, dow : Word;

Function adconv(channel: integer): integer;
var del : integer;
begin

{ Start ADC = 0, Ale ADC = 0, A0-2 ADC = bit 0-3 di channel }
port[portb] := (port[portb] and $E0 ) or (channel and $7);
del := del + del; { Ritardo stabilizz. out 8255 }

port[portb] := port[portb] or $10; { Ale ADC = 1 }
del := del + del; { Ritardo stabilizz. out 8255 }
port[portb] := port[portb] and $EF; { Ale ADC = 0 }
del := del + del; { Ritardo stabilizz. out 8255 }

port[portb] := port[portb] or $8; { Start ADC = 1 }
del := del + del; { Ritardo stabilizz. out 8255 }
port[portb] := port[portb] and $F7; { Ale ADC = 0 }

for del := 0 to 100 do ; { Attesa inizio conversione }

while (port[portc] and 1) = 0 do ; { Attesa End Of Conversion }
```

```

adconv := port[porta];           { Lettura dato convertito }

end;

{ La funzione T1 legge il canale 0 dell'ADC e lo trasforma in temperatura
  misurata in °C }
Function T1 : real;

  Var
  A : integer;
  Begin
  A:=ADCONV(0);
  T1 := A*300.0/255;
  End; { T1 }

{ La funzione T2 legge il canale 1 dell'ADC e lo trasforma in temperatura
  misurata in °C }
Function T2 : real;

  Var
  A : integer;
  Begin
  A:=ADCONV(1);
  T2 := A*300.0/255;
  End; { T2 }

procedure square(x, y, x1, y1, s, t: integer);
var i : integer;
begin
  { textbackground(s); }
  textcolor(t);
  gotoxy(x,y);

  write('U');
  for i := x+1 to x+x1-2 do
    write('A');
  write(';');
  for i:= 1 to y1-2 do
    begin
      gotoxy(x, y+i);
      write("³");
      gotoxy(x+x1-1, y+i);
      write("³");
    end;

    gotoxy(x, y+y1-1);
    write('A');

    for i := x+1 to x+x1-2 do
      write('A');

      gotoxy(x+x1-1, y+y1-1);
      write('U');
end;

procedure dsquare(x, y, x1, y1, s, t: integer);
var i : integer;
begin
  { textbackground(s); }
  textcolor(t);
  gotoxy(x,y);

```

```

write('É');
for i := x+1 to x+x1-2 do
  write('Í');

write('»');
for i:= 1 to y1-2 do
  begin
    gotoxy(x, y+i);
    write("°");
    gotoxy(x+x1-1, y+i);
    write("°");
  end;

gotoxy(x, y+y1-1);
write('É');

for i := x+1 to x+x1-2 do
  write('Í');

gotoxy(x+x1-1, y+y1-1);
write("¼");
end;

Procedure Quadro;
var i,j : integer;
Begin
  textbackground(blue);
  For i:=1 to 80 do
    for j:=1 to 25 do write(' ');

  dsquare(2, 1, 78, 25, blue, yellow);
  square(3, 2, 76, 3, blue, green); { controllo temperatura industriale }

  square(3,5,37,3, blue, white); { T1 predisposta }
  square(40,5,39,3, blue, white); { T2 predisposta }

  square(3,8,37,3, blue, white); { T1 % }
  square(40,8,39,3, blue, white); { T2 % }

  square(3,11,37,3, blue, white); { T1 misurata }
  square(40,11,39,3, blue, white); { T2 misurata }

  square(3,14,37,3, blue, white); { T1 min }
  square(40,14,39,3, blue, white); { T2 min }

  square(3,17,37,3, blue, white); { T1 Max }
  square(40,17,39,3, blue, white); { T2 Max }

  square(3,20,37,4, blue, white); { rel, ed allarme 1 }
  square(40,20,39,4, blue, white); { rel, ed allarme 1 }

  gotoxy(23,3); write('CONTROLLO DI TEMPERATURA INDUSTRIALE');
  gotoxy(4,6); write('T1 predisposta ');
  gotoxy(41,6); write('T2 predisposta ');
  gotoxy(4,9); write('% T1 ');
  gotoxy(41,9); write('% T2 ');
  gotoxy(41,12); write('T2 misurata ');
  gotoxy(4,12); write('T1 misurata ');
  gotoxy(4,15); write('T1 min ');
  gotoxy(41,15); write('T2 min ');
  gotoxy(4,18); Write('T1 Max ');

```

```

gotoxy(41,18); write('T2 Max ');
gotoxy(4,21); Write('rel, 1');
gotoxy(41,21); write('rel, 2');
gotoxy(4,22); write('allarme1 ');
gotoxy(41,22); write('allarme2 ');
gotoxy(37,24); TextColor(magenta);
                write('by Comini - Rizzinelli - Varisco 5BI 96-97');
                TextColor(white);

```

End;

Procedure Allarme;

begin

```
gotoxy(14,22); textcolor(green);Write(Tal1:3:0); Textcolor(white);
```

```
If mt1 < Tal1 then
```

```
begin
```

```
port[portb] := port[portb] or $20;    { Allarme1 off }
```

```
gotoxy(21,22); write('OFF ');
```

```
Allarme1 := False;
```

```
end
```

```
else begin
```

```
port[portb] := port[portb] And $DF; { Allarme1 on }
```

```
gotoxy(21,22);
```

```
Allarme1 := True;
```

```
TextColor(red+blink);
```

```
write('ON ');
```

```
TextColor(white);
```

```
end;
```

```
gotoxy(51,22); textcolor(green);write(tal2:3:0); Textcolor(white);
```

```
If mt2 < Tal2 then
```

```
begin
```

```
port[portc] := port[portc] or $80;    { Allarme2 off }
```

```
gotoxy(58,22);write('OFF');
```

```
Allarme2 := False;
```

```
end
```

```
else begin
```

```
port[portc] := port[portc] And $7F;  { Allarme2 on }
```

```
gotoxy(58,22);
```

```
Allarme2 := True;
```

```
TextColor(red+blink);
```

```
write('ON ');
```

```
TextColor(white);
```

```
end;
```

end;

Procedure rele; { i rel \bar{S} come i led vengono attivati in logica negativa }

Begin

```
If mt1 > T1min Then
```

```
Begin
```

```
port[portb] := port[portb] or $80;    { rele1 off }
```

```
gotoxy(21,21); write('OFF');
```

```
End;
```

```
If mt1 < T1max Then
```

```
Begin
```

```
port[portb] := port[portb] and $7F;  { rele1 on }
```

```
gotoxy(21,21);
```

```
textcolor(yellow);
```

```
write('ON ');
```

```
textcolor(white);
```

```

End;

If mt2 > T2min Then
  Begin
    port[portb] := port[portb] or $40;    { rele2 off }
    gotoxy(58,21); write('OFF');
  End;

If mt2 < T2max Then
  Begin
    port[portb] := port[portb] and $BF;    { rele2 on }
    gotoxy(58,21);
    textcolor(yellow);
    write('ON ');
    textcolor(white);
  End;
End;

BEGIN { PRINCIPALE }
  Tal1 := 250; Tal2 := 280;
  clrscr;    { cancello lo schermo }
  Quadro;

  GetDate(y,m,d,dow);
  Gotoxy(4,24); Write(days[dow],', ',d:0,'/', m:0,'/', y:0);

  port[control] := $91;    { PA=INP, PB=OUT, PC0-3=INP, PC4-7=OUT }
  port[portb] := $E0;    { Uscire PB = OFF }
  port[portc] := $80;

  gotoxy(21,6); Readln(Temp1);
  gotoxy(21,9); Readln(p1);
  T1min :=Temp1*(1-p1/100);
  gotoxy(21,15); write(T1min:6:2);
  T1max :=Temp1*(1+p1/100);
  gotoxy(21,18); write(t1max:6:2);

  gotoxy(58,6); Readln(Temp2);
  gotoxy(58,9); Readln(p2);
  T2min :=Temp2*(1-p2/100);
  gotoxy(58,15); write(T2min:6:2);
  T2max :=Temp2*(1+p2/100);
  gotoxy(58,18); write(t2max:6:2);

  repeat
  begin
    mt1 := T1;
    mt2 := T2;
    gotoxy(21,12);
    if allarme1
      then textcolor(red)
      else textcolor(white);
    write(mt1:6:2);
    gotoxy(58,12);
    if allarme2
      then textcolor(red)
      else textcolor(white);
    write(mt2:6:2);
  allarme;
  rele;
  end

```

```
    until FALSE;  
end.
```
