

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

Progetto di un Parcheggio Auto Computerizzato

Realizzato da :

BELLERI SIMONE

CAPOZZI ANTONIO

ZILIANI MANUEL

classe 5BI TIEE

corso per Tecnici delle Industrie Elettriche ed Elettroniche

anno scolastico 1996-97

1. DISEGNO INIZIALE PARCHEGGIO	3
2. SPIEGAZIONE GENERALE DELLA PROVA	3
3. INTERFACCIA SPERIMENTALE PER COMPUTER	4
4. PROGETTO GENERALE INTERFACCIA	6
INTERFACCIA PERIFERICA PROGRAMMABILE 82C55	7
<i>DATA BUS BUFFER</i>	8
<i>READ / WRITE and CONTROL LOGIC</i>	8
MODALITÀ DI FUNZIONAMENTO I/O	9
<i>HANDSHAKE</i> o “ <i>stretta di mano</i> ”	10
4. TASTIERINO NUMERICO	11
5. APPUNTI SU TURBO PASCAL	15

1. Disegno iniziale parcheggio

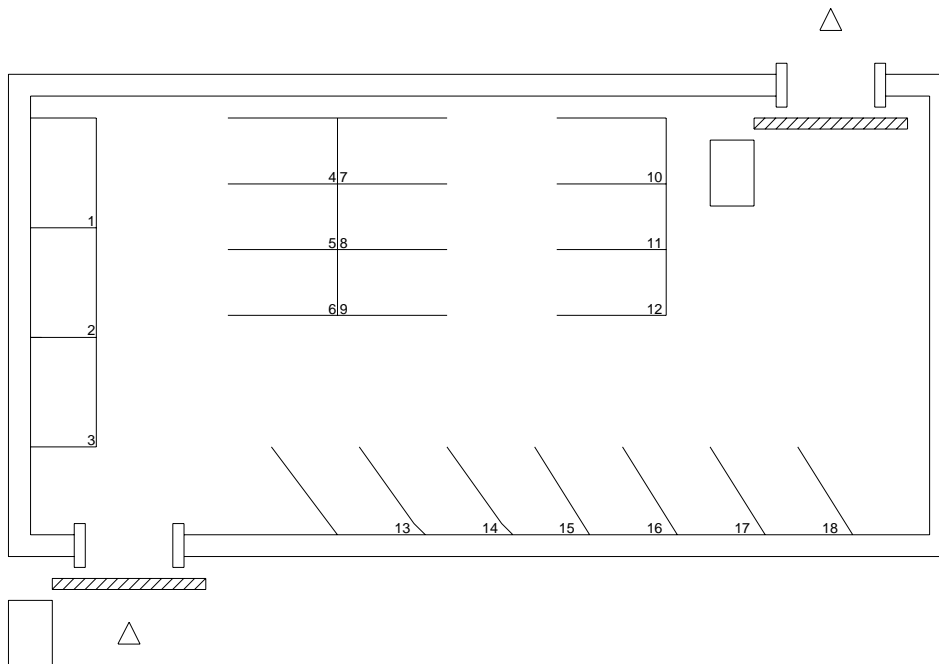


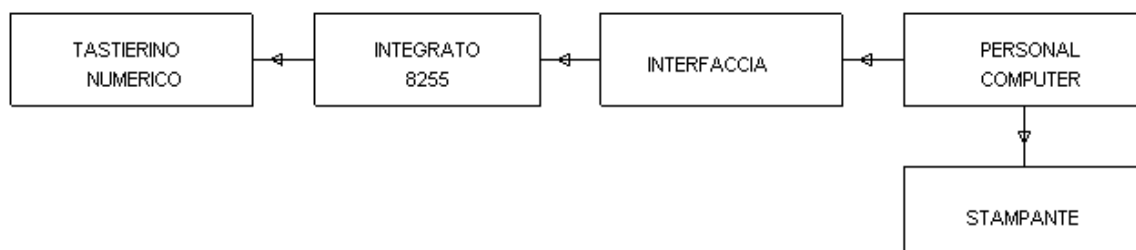
fig.1

2. Spiegazione generale della prova

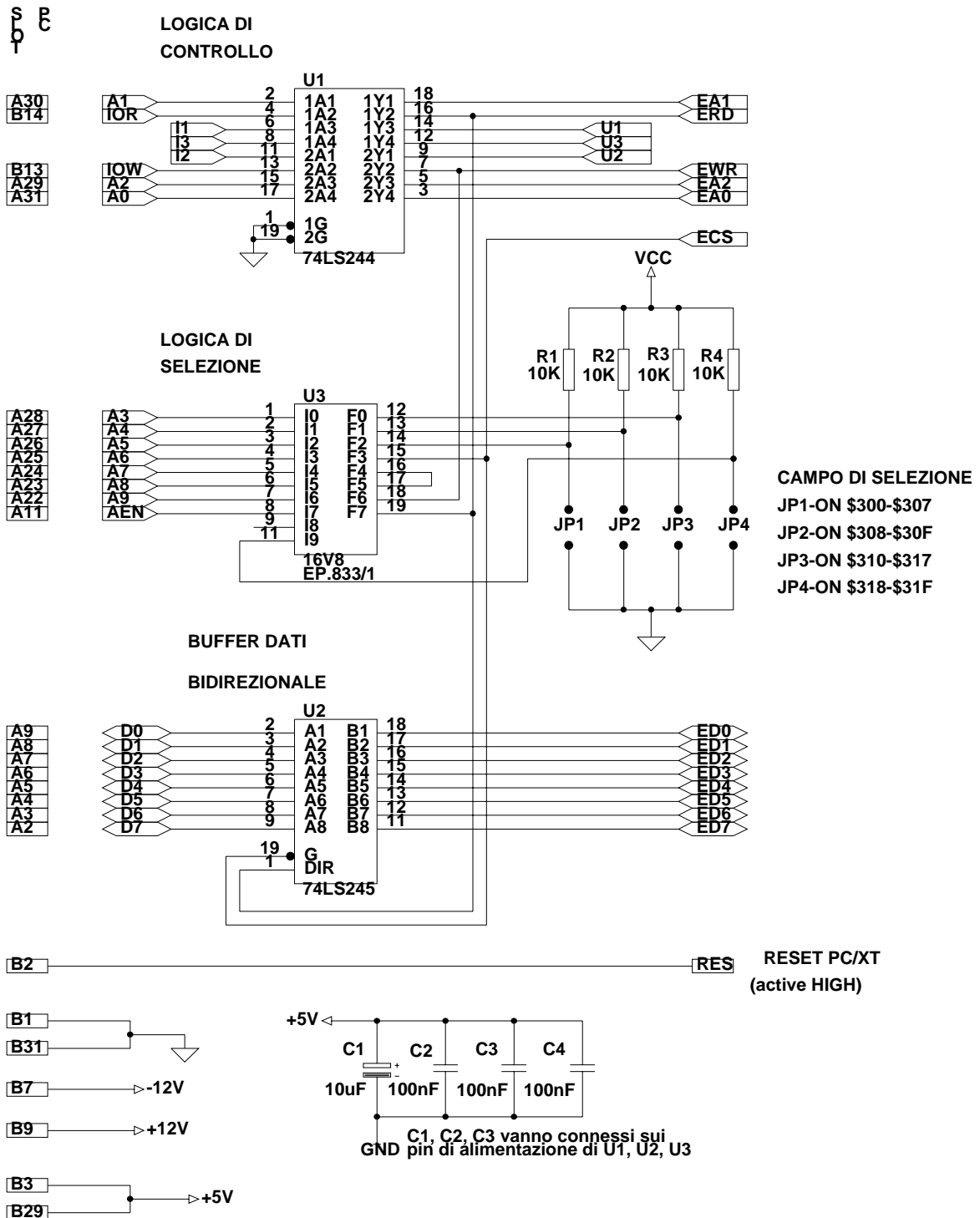
La relazione da noi eseguita è un progetto teorico/pratico di un garage pubblico per automobili predisposto per contenere un numero limitato di vetture ; il nostro progetto (come si vede in figura 1) comprende due sbocchi , uno di entrata che serve alle vetture per accedere al garage , ed uno di uscita che serve alle vetture per lasciare il garage. Lo sbocco di entrata sarà compreso di una sbarra che fermerà le vetture ; esse saranno costrette a ritirare un biglietto dall'apposita biglietteria dove verrà scritto l'ora e i minuti cui la vettura è entrata nel garage. Una volta che il conducente avrà ritirato il biglietto la sbarra si dovrà alzare. Per gli abbonati al parcheggio invece ci sarà tramite l'uso di un "tastierino", la possibilità di comporre un codice personale per ogni individuo che gli permetterà di non pagare .

Anche in uscita avremo una sbarra che non permetterà al cliente del parcheggio di uscire senza aver pagato; la sbarra di entrata si dovrà alzare automaticamente dopo che ogni cliente avrà ritirato il biglietto con il quale successivamente dovrà andare a pagare. La sbarra di uscita invece non sarà automatica, ma manuale e sarà alzata e abbassata da una persona apposita alla quale precedentemente si sarà pagato la cifra prestabilita , che sarà proporzionale al tempo che la vettura è restata nel parcheggio.

Per la realizzazione di questo progetto è stato usato un personal computer Amstrad con relativa stampante interfacciato ad un “tastierino” numerico. Il personal computer è stato programmato con l’utilizzo del programma Turbo Pascal in Bolard .



3. Interfaccia sperimentale per computer



L' interfaccia che abbiamo utilizzato nella nostra prova , come si può vedere in figura , è formata da una scheda “mille fori” con inseriti tre integrati:

IC1 = buffer di controllo linea

IC2 = transcodifica

IC3 = buffer bidirezionale per i dati

Sulla superficie di tale scheda potranno poi essere inseriti tutti i circuiti sperimentali che l'utente desidera , utilizzando logicamente integrati o transistor , (nel nostro caso l'integrato 8255).

Ritornando al nostro circuito precisiamo che il rettangolo riportato sul lato sinistro è il connettore d'innesto per il Bus; le sigle presenti all' interno di tale rettangolo rappresentano la numerazione stampigliata sul connettore del Bus, mentre le sigle al di fuori di tale rettangolo indicano il segnale presente su tale connessione : A0-A1-A2 = primi tre bit del bus degli indirizzi.

4. Progetto generale interfaccia

IOR = abbreviazione di input/output read , che significa lettura di un dispositivo ingresso uscita , cioè la lettura dei dati avverrà se su questo pin avremo un livello logico 0 .

IOW = abbreviazione di input/output write , che significa scrittura di un dispositivo ingresso uscita , cioè la scrittura dei dati avverrà se su questo pin avremo un livello logico 0.

A3-A9 = bit del bus degli indirizzi

AEN = un livello logico 0 su questo pin , significa che l' indirizzo presente in quell'istante sul bus è valido.

+5 volt = tensione di alimentazione della scheda

D0 .. D7 = sono gli otto bit dei bus

+12 e -12 volt = tensione stabilizzata che potremo utilizzare per alimentare circuiti sperimentali montati sulla scheda.

Detto questo , possiamo descrivere le funzioni svolte dai tre integrati presenti in tale scheda:

74LS 44: questo integrato IC1 serve da buffer per i segnali provenienti dal bus degli indirizzi, per i segnali di lettura o scrittura dei dati , in modo da separare le linee dei bus principale dai segnali portati all' esterno sulla scheda sperimentale; sui tre piedini liberi di ingresso potremo applicare il segnale di reset, di clock, oppure uno dei segnali di interruzione , o altri ancora .

EP 833 : questo secondo integrato , siglato IC2, è una prom programmata utile per decodificare gli indirizzi ; sui piedini di uscita (14-13-12e11), abbiamo collocato un piccolo connettore (vedi sul connettore J1 i terminali siglati A B C D) che ponticellate , ci permetteranno di selezionare quattro diversi gruppi di indirizzi da assegnare alla scheda sperimentale . Ognuno di questi gruppi è composto da otto indirizzi e , pertanto , avremo a disposizione un totale di $4 * 8 = 32$ indirizzi.

Dalle combinazioni che abbiamo, potremo ottenere una mappa di indirizzamento divisa in quattro gruppi , che corrisponderanno ciascuno a quattro ponticelli del connettore J1. In particolare :

- - ponticello J1 su A : indirizzo esadecimale della scheda da 300 a 307 , che corrisponde in decimale da 768 a 775.
- - ponticello J1 su B : indirizzo esadecimale della scheda da 308 a 30F , che corrisponde in decimale da 776 a 783.
- - ponticello J1 su C : indirizzo esadecimale della scheda da 310 a 317 , che corrisponde in decimale da 784 a 791.
- - ponticello J1 su D : indirizzo esadecimale della scheda da 318 a 31F , che corrisponde in decimale da 792 a 799.

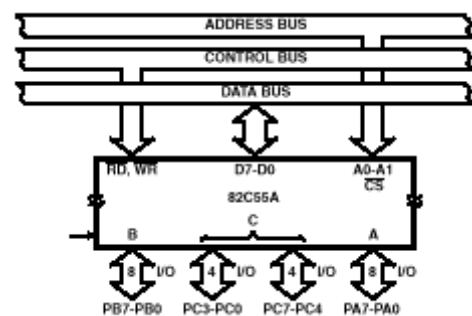
SN74L245 : l' integrato siglato IC3 , come è facile intuire , verrà utilizzato per trasferire gli otto bit dei dati della scheda sperimentale al bus del computer e viceversa , dal bus del computer alla scheda.

Una volta detto questo , potremo cominciare a realizzare la scheda montando su essa i tre integrati con i loro rispettivi zoccoli , cercando nella fase di saldatura di non creare cortocircuiti tra i pin ; di seguito potremo iniziare a inserire nella scheda gli altri componenti presenti sullo schema elettrico

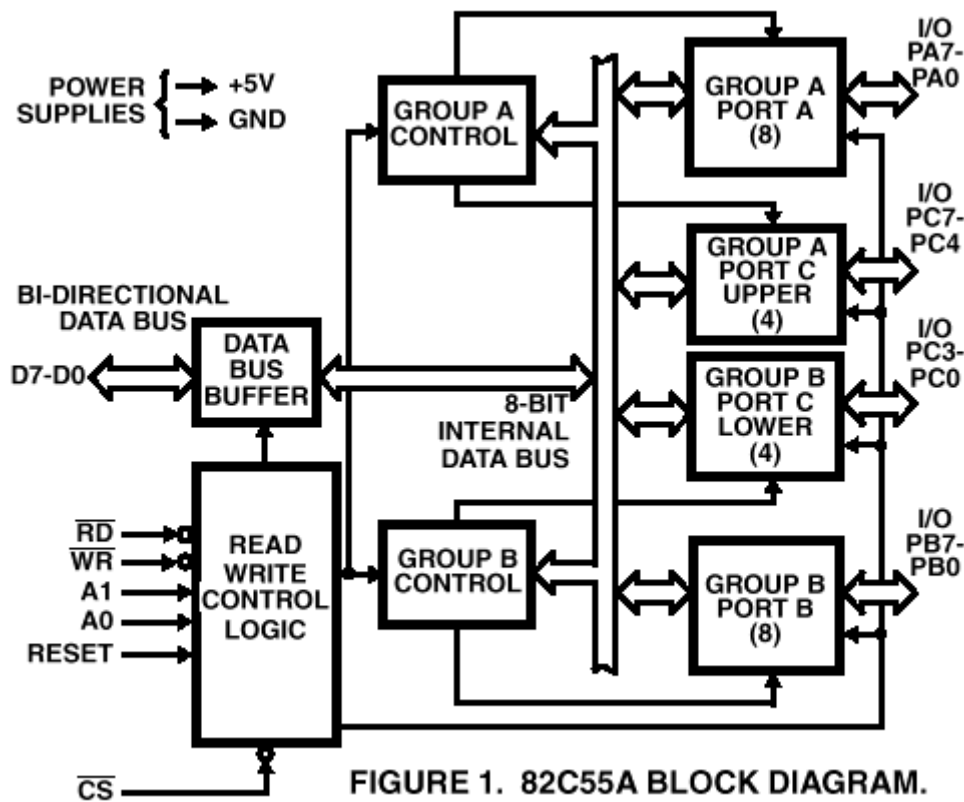
Interfaccia periferica programmabile 82C55

L' 8255A é un dispositivo di interfacciamento periferico programmabile, fabbricato per essere usato con microprocessori della famiglia INTEL. La sua funzione é 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 é 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.

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

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.

4. Tastierino numerico

Il disegno sotto eseguito in figura rappresenta il circuito elettrico del tastierino numerico usato nella nostra prova , per gli utenti del parcheggio , che possiedono un abbonamento ad esso. Il circuito elettrico del tastierino è composto da 4 diodi 1N4148 , da 16 pulsanti , da 4 resistenze di pull up e in uscita da 4 porte NOT triggered (Schmit) sui quali più avanti approfondiremo il discorso. Come si vede dalla figura il nostro circuito avrà in entrata l’uscita dell’interfaccia 8255, e in uscita l’entrata dell’ 8255. Per la gestione del tastierino numerico, inserito nella biglietteria di entrata del parcheggio , è stata usata una programmazione in linguaggio Pascal.

```
Function Tastiera:char;
const
  tabella_uscita:array[1..16] of char =( '0','1','2','3',
    '4','5','6','7',
    '8','9','A','B',
    'C','D','E','F');
  tabella_entrata:array[1..16] of integer=(1,2,4,8,
    11,12,14,18,
    21,22,24,28,
    31,32,34,38);
var i,t : integer ;
var tas,ttt:char;
    line : array[1..4] of integer;
Begin
  Tas := chr(0);

  { Ricavo i codici di scansione della tastiera }

  PORT[portc] := $E;      { PC0 = 0 (select riga 0 )
  t := t + t;             { Ritardo attesa stabilizzaz. out 8255 }
  line[1] := (PORT[portc] and $F0) div 16; { Legge I gruppo dati }

  PORT[portc] := $D;      { PC1 = 0 (select riga 1 )
  t := t + t;             { Ritardo attesa stabilizzaz. out 8255 }
  line[2] := (PORT[portc] and $F0) div 16; { Legge II gruppo dati }
```

```

PORT[portc] := $B;      { PC2 = 0 (select riga 2 )
t := t + t;           { Ritardo attesa stabilizzaz. out 8255 }
line[3] := (PORT[portc] and $F0) div 16; { Legge III gruppo dati }

PORT[portc] := $7;      { PC3 = 0 (select riga 3 )
t := t + t;           { Ritardo attesa stabilizzaz. out 8255 }
line[4] := (PORT[portc] and $F0) div 16; { Legge IV gruppo dati }

```

{ Analizzo i codici di scansione per eliminare codici derivanti dalla pressione simultanea di più tasti }

```

i:=0;
for k:=1 to 4 do
begin
if line[k] <>0 then
begin
i:=i+1;
if (line[k] in [1,2,4,8]) Then
begin
line[k]:=(k-1)*10+line[k];
end
else line[k] :=0;
end;
end;
end;

```

{ Associa al codice di scansione la codifica ASCII del tasto premuto }

```

if i=1 then
begin
for k:=1 to 4 do
begin
if line[k] <>0 then
begin
for j :=1 To 16 do
begin
if line[k]=tabella_entrata[j] then
Tas := tabella_uscita[j];
end;
end;
end;
end;
end;
end;

```

{ se tastiera premuta attendo rilascio }

```

if Tas <> Chr(0) Then
Begin
Port[Portc] := 0;
delay(100);
Repeat
Delay(100);
Until (Port[Portc] and $F0) = 0;
Port[Portc] := $F ;
Delay(100);
End;
Tastiera := Tas;
End; { Tastiera }

```

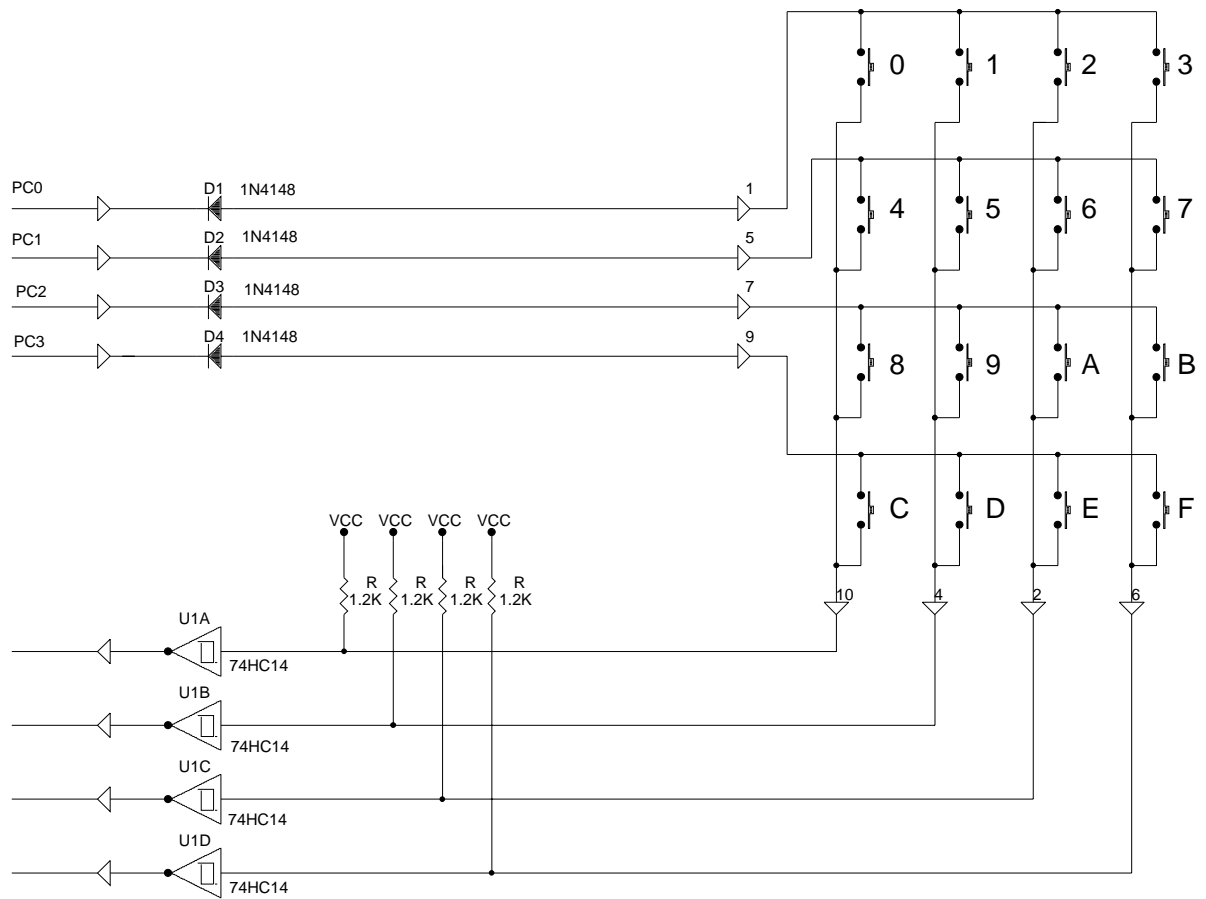
La funzione attraverso la quale viene effettuata la lettura del tastierino utilizza tre array (vettori matematici) denominati Tabelle_Uscita da 16 elementi, Tabella_entrata da 16 elementi e line da 4

elementi. All'interno del primo array sono collocati nell'ordine i codici ASCII corrispondenti al dato associato al pulsante (codice ASCII dello "0" al tasto 0, codice ASCII della lettera F in al tasto F); nel secondo array sono collocati i codici ammessi e corrispondenti alla pressione dei 16 tasti (premuti singolarmente). All'interno dell'array line sono contenuti i codici derivanti dalla scansione del tastierino ; scansione ottenuta attivando una alla volta le linee connesse al Port C (parte bassa da PC0 a PC3).

Per poter collegare il nostro tastierino con l' integrato 8255 , che è costituito da 3 port, *port A* , *port B* e *portC*, dei quali sul primo essendo composto da soli input possiamo solo scrivere, il secondo è composto da soli output e potremo perciò solo leggere , mentre il port C essendo composto da input e output , potremo sia leggere che scrivere. Essendo il port C diviso in input ed output , abbiamo dovuto dare dei valori per far attivare le varie righe (es: \$E = riga 0 ; \$D = riga 1 ; ecc.) e successivamente per ogni abbiamo inserito un ritardo di stabilizzazione.

Successivamente il programma ci controllerà se ogni line corrisponde al valore 1-2-4-8 ; se la line corrisponderà ad uno di questi valori elencati attraverso un calcolo " logico " ci darà il tasto che è stato premuto. Se verranno premuti due stati contemporaneamente il valore in uscita dovrà essere 0 .

I codici in entrata sono composti da quattro caratteri e ogni carattere può assumere un valore che può variare da 0 a 9. Se un abbonato al nostro parcheggio ha un codice minore di quattro caratteri (es : 2-3 ecc.) alla fine dovrà digitare il tasto F per concludere l' operazione .



INTEGRATO 74HC14

Questo integrato denominato 74HC14, è costituito da 6 porte NOT di tipo triggered (trigger di Schmitt) indipendenti caratterizzate dalla funzione booleana $Y = \overline{A}$. L'integrato dispone di 14 pin (piedini), divisi in pin di alimentazione (pin 14 VCC 5 Volt, pin 7 GND), da pin di entrata (pin 1-3-5-9-11-13) e da pin di uscita (pin 2-4-6-8-10-12); ogni entrata sarà collegata alla sua uscita e perciò il pin 1 di entrata sarà collegato al suo pin 2 di uscita e così via. Il 74HC14 potrà lavorare in condizioni normali da -40 a +85 gradi centigradi; la caratteristica principale sarà comunque quella che in uscita avremo il valore logico negato dell'entrata.

5. Appunti su turbo pascal

Queste brevi note vogliono essere solamente una sintesi delle nozioni essenziali di cui deve disporre un programmatore che si appresta ad utilizzare Turbo Pascal. La programmazione degli elaboratori può essere attuata facendo uso del linguaggio macchina oppure di linguaggi di programmazione detti ad alto livello; il linguaggio macchina è un linguaggio specifico di una determinata CPU e quindi presuppone una buona conoscenza del set di istruzioni, delle modalità di esecuzione e dell'architettura del sistema utilizzato. Tra i linguaggi ad alto livello uno dei più conosciuti è quello PASCAL adatto a risolvere sia problemi di tipo scientifico, che problemi di altro genere. Il programma scritto dal programmatore ed introdotto nella memoria dell'elaboratore viene detto SORGENTE; quello materialmente eseguito dalla CPU e perciò codificato in linguaggio macchina viene denominato programma OGGETTO. Sarà perciò necessario disporre di un traduttore che partendo da istruzioni redatte in linguaggio ad alto livello le trasformi in equivalenti istruzioni stese in linguaggio macchina.

Il compito del programmatore consiste nel:

- Introdurre le informazioni nel programma - **input**
- Creare uno spazio in cui memorizzarle - **dati**
- Usare le istruzioni giuste per manipolarle - **operazioni**
- Fornire all'utente le risposte - **output**

Queste istruzioni possono essere organizzate in diversi modi:

- Alcune vengono eseguite solo se una o più condizioni specificate sono vere - **esecuzione condizionale**
- Altre vengono ripetute un certo numero di volte - **cicli**
- Altre vengono frazionate in blocchi separati che possono essere eseguiti in punti differenti del programma - **subroutine**

Ecco una breve spiegazione dei sette elementi fondamentali:

Input

Significa leggere i dati dalla tastiera, da disco o da una porta di I/O.

Dati

Sono le costanti, le variabili e le strutture che contengono numeri (interi e reali), testo (caratteri e stringhe) o indirizzi (di variabili e strutture).

Operazioni

Servono ad assegnare o calcolare valori, (addizione, divisione, ecc.) e ad effettuare confronti (uguale a, diverso da, ecc.).

Output

Significa scrivere informazioni su schermo, su disco o in una porta di I/O.

Esecuzione condizionale

Significa eseguire un insieme di istruzioni solo se una certa condizione è vera (quindi saltando tal istruzioni o eseguendone altre se la condizione è falsa) oppure quando un valore ricade all'interno di un intervallo stabilito.

Cicli

Servono per eseguire un insieme di istruzioni ripetutamente cioè o un numero di volte prestabilito o mentre una condizione è vera o finchè una condizione è falsa.

Subroutine

E' un insieme di istruzioni cui è stato assegnato un nome e può essere eseguita in più punti del programma facendo semplicemente riferimento al nome.

TIPI DI DATI

Quando scriviamo un programma, si ha a che fare, generalmente, con cinque tipi fondamentali di informazioni : **numeri interi, numeri reali, caratteri e stringhe, espressioni Booleane e puntatori.**

- **Interi (Integer)** : sono i numeri interi usuali (es. 1, 5, -21,7).
- **Reali (Real)** : hanno una parte frazionaria (3.14159) ed un esponente (2.579×10^{E24}). Sono anche noti come numeri in virgola-mobile.
- **Caratteri (Character)** : sono costituiti dalle lettere dell'alfabeto, da simboli e da numeri 0-9. Possono essere usati individualmente (a, Z, 1, 3) o a gruppi nelle stringhe di caratteri ('Questa è una stringa').
- **Espressioni Booleane (Boolean)** : possono produrre solo due valori vero o falso. Vengono utilizzate nelle espressioni condizionali.
- **Puntatori (Pointer)** : contengono indirizzi di locazioni di memoria del computer, le quali, a loro volta, contengono delle informazioni.

Per spiegare meglio i tipi di dati interi e reali possiamo rappresentarli in due tabelle, che contengono il tipo, gli intervalli, e la dimensione in byte.

Tabella 1 : Tipi di dati INTERI

TIPO	INTERVALLO	DIMENSIONE IN BYTE
Byte	0 .. 255	1
Shortint	-128 .. 127	1
Integer	-32768 .. 32767	2
Word	0 .. 65535	2
Longint	-2147483648 .. 2147483647	4

Tabella 1 : Tipi di dati REALI

TIPO	INTERVALLO	CIFRE SIGNIF.	DIM. IN BYTE
Real	$2.9 \times 10E-39$.. $1.7 \times 10E38$	11 - 12	6
Single	$1.5 \times 10E-45$.. $3.4 \times 10E38$	7 - 8	4
Double	$5.0 \times 10E-324$.. $1.7 \times 10E308$	15 - 16	8
Extended	$1.9 \times 10E-4951$.. $1.1 \times 10E4932$	19 - 20	10
Comp*	$-2E+63+1$.. $2E+63-1$	19 - 20	8

Programma Sorgente in Linguaggio PASCAL 6.0

```

program parcheggio(input,output);

uses DOS,Crt,Printer,bios;
const
  days: array [0..6] of string [9]=('Domenica','Lunedì','Martedì',
    'Mercoledì','Giovedì','Venerdì',
    'Sabato');

  Abb : array [1..10] of integer=(9777,1951,6372,2931,4553,
    6990,3482,4707,7423,176);

  scheda=$300;
  porta=scheda+$0;   { solo input }
  portb=scheda+$1;   { solo output }
  portc=scheda+$2;   { PC0-PC3=OUTPUT PC4-PC7=INPUT }
  control=scheda+$3; { registro di controllo solo write }

  dport = $378;      { Data-Port stampante parallela }
  sport = $379;      { Status-Port stampante parallela }

type no_abb=array[1..100,1..5] of word;
var
  y,m,d,dow,imp: word;
  h,m1,s,hund,mintot:word;
  x:no_abb;
  i,j,k,l,n,w,z:integer;
  tasto,c:char;
  cont,num,num1,cont2:integer;
  a:char;

procedure dsquare(x, y, x1, y1, s, t: integer);
var i : integer;
begin

```

```

textbackground(s);
textcolor(t);
gotoxy(x,y);
Write(Chr(201)); { riquadro superiore SX }
for i := x+1 to x+x1-2 do Write(Chr(205)); { doppia linea orizzontale }

write(Chr(187)); { riquadro superiore DX }
for i:= 1 to y1-2 do
  begin
    gotoxy(x, y+i);
    write(chr(186)); { doppia linea verticale }
    gotoxy(x+x1-1, y+i);
    write(chr(186)); { doppia linea verticale }
  end;

gotoxy(x, y+y1-1);
write(chr(200)); { riquadro inferiore sinistro }

for i := x+1 to x+x1-2 do Write(Chr(205)); { doppia linea orizzontale }

gotoxy(x+x1-1, y+y1-1);
write(chr(188)); { riquadro inferiore destro }
end;

procedure ritardo(i:integer);
var t:integer;
begin
  for t:=1 to i do;
end;

procedure menu;
begin
  gotoxy(45,6);
  Write('MENU TASTIERINO');
  gotoxy(45,8);
  Write('Non abbonati E');
  gotoxy(45,10);
  Write('Abbonati A');
  gotoxy(45,12);
  write('Uscita vettura D');
  gotoxy(45,14);
  write('Uscita in dos 0');
end;

Function Tastiera:char;
const
tabella_uscita:array[1..16] of char =( '0','1','2','3',
'4','5','6','7',
'8','9','A','B',
'C','D','E','F');
tabella_entrata:array[1..16] of integer=(1,2,4,8,
11,12,14,18,
21,22,24,28,
31,32,34,38);

var i,t : integer ;
var tas,ttt:char;
line : array[1..4] of integer;
Begin
  Tas := chr(0);

  PORT[portc] := $E; { PC0 = 0 (select riga 0 )

```

```

t := t + t;          { Ritardo attesa stabilizzaz. out 8255 }
line[1] := (PORT[portc] and $F0) div 16; { Legge I gruppo dati }

PORT[portc] := $D;   { PC1 = 0 (select riga 1 ) }
t := t + t;          { Ritardo attesa stabilizzaz. out 8255 }
line[2] := (PORT[portc] and $F0) div 16; { Legge II gruppo dati }

PORT[portc] := $B;   { PC2 = 0 (select riga 2 ) }
t := t + t;          { Ritardo attesa stabilizzaz. out 8255 }
line[3] := (PORT[portc] and $F0) div 16; { Legge III gruppo dati }

PORT[portc] := $7;   { PC3 = 0 (select riga 3 ) }
t := t + t;          { Ritardo attesa stabilizzaz. out 8255 }
line[4] := (PORT[portc] and $F0) div 16; { Legge IV gruppo dati }

```

```

i:=0;
for k:=1 to 4 do
begin
  if line[k] <>0 then
  begin
    i:=i+1;
    if (line[k] in [1,2,4,8]) Then
    begin
      line[k]:=(k-1)*10+line[k];
    end
    else line[k] :=0;
  end;
end;

if i=1 then
begin
  for k:=1 to 4 do
  begin
    if line[k] <>0 then
    begin
      for j :=1 To 16 do
      begin
        if line[k]=tabella_entrata[j] then
          Tas := tabella_uscita[j];
        end;
      end;
    end;
  end;
end;

```

```

{ se tastiera premuta attendo rilascio }
if Tas <> Chr(0) Then
Begin
  Port[Portc] := 0;
  delay(100);
  Repeat
  Delay(100);
  Until (Port[Portc] and $F0) = 0;
  Port[Portc] := $F ;
  Delay(100);
End;
Tastiera := Tas;
End; { Tastiera }

```

```

Function codice(protezione:boolean):integer;
Var K,z,t : integer;

```

```

    st : string[4];
    a : char;
Begin
codice:=0;
k:=0; {contatore caratteri}
st:=""; {stringa}
repeat
a:=tastiera;
if (a>='0') and (a<='9') then
begin
k:=k+1;
if protezione Then write('*')
Else write(a);
st:=st+a;
end;
until (k=4) or (a='F');
val(st,t,z);
codice := t;
end;

```

```

Function Stampa : Boolean;
var a : byte;

```

```

Begin
stampa := false;
a := port[sport] And $FE;
case a of
$D8 : Begin
stampa := true;
gotoxy(4,22);
write(' ');
End;
$50 : Begin
gotoxy(4,22);
Textcolor(red+blink);
write('Stampante Off line ');
Textcolor(white);
End;
$00 : Begin
gotoxy(4,22);
Textcolor(red+blink);
write('Stampante spenta ');
Textcolor(white);
End;
$78 : Begin
gotoxy(4,22);
Textcolor(red+blink);
write('Cavo stampante staccato');
Textcolor(white);
End;
$70 : Begin
gotoxy(4,22);
Textcolor(red+blink);
write('Carta mancante ');
Textcolor(white);
End;
Else Begin
gotoxy(4,22);
Textcolor(red);
write(hex(a):2,' Codice strano ');
Textcolor(white);

```

```

        End;

End;

End;

BEGIN { PROGRAMMA PRINCIPALE }
PORT [CONTROL] :=$8A;
ClrScr;

textbackground(blue);
for i:=1 to 80 do
begin
for j:=1 to 25 do write(' ');
end;
dsquare(2, 1, 78, 25, blue, white);
gotoxy(20, 3);

write('PARCHEGGIO AUTO BELLERI CAPOZZI ZILIANI');

For I:=1 TO 100 DO
BEGIN
for j:=1 to 4 do x[i,j] := 0;
End;
i:=0; j:=1;
cont:=0;
mintot:=0;
GetDate(y,m,d,dow);
gotoxy(45,5);
write(days[dow],',',D:0,',',M:0,',',y:0);
REPEAT
Menu;
gotoxy(45,16);
Write('Introdurre scelta ');
gotoxy(66,16);
Write(' ');
gotoxy(66,16);

Repeat
c:=Tastiera;
Until c<> chr(0);

c:=upcase(c);
Case c of
'E' : Begin
if cont<10 then
begin
I:=I+1; { contatore macchine entranti }
GetTime(h,m1,s,hund);
gotoxy(4,10);
write('Orario di entrata : ',h,',',m1,',',s);
cont := cont+1; { contatore macchine presenti }
gotoxy(4,14);
write('VETTURE PRESENTI: ',cont:2);
x[i,1]:=h;
x[i,2]:=m1;
x[i,3]:=s;
x[i,5]:=1;
gotoxy(25,6);
write(' ');
if stampa then

```

```

Begin
  Writeln(LST,'Parcheggio Belleri Capozzi Ziliani & c. S.p.A. ');
  Writeln(LST);
  Writeln(LST,'Vettura N :',i);
  Writeln(LST);
  Writeln(LST,days[dow],',',D:0,',',M:0,',',y:0);
  Writeln(LST);
  Writeln(LST,'Orario di entrata : ',h:',',m1:',',s);
  Writeln(LST);
End;
end
else
  Begin
    gotoxy(25,6);
    write ('PARCHEGGIO PIENO ',#7);
  End;
  gotoxy(45,18);
  write(' ');
End;
'D' :begin
  if cont<>0 then
    begin
      gotoxy(45,18);
      write(' ');
      repeat
        gotoxy(45,20);
        write('Numero della vettura ');
        gotoxy(66,20);
        write(' ');
        gotoxy(66,20);
        num := codice(false) ;
      until (x[num,5]=1) And (num <= I);
      delay(2000);
      gotoxy(45,20);
      write(' ');
      GetTime(h,m1,s,hund);
      mintot:=(m1*60+s)-(x[num,2]*60+x[num,3]);
      If mintot<60 then x[num,4]:=200
      else
        begin
          x[num,4]:=(mintot div 60)*200;
          if (mintot mod 60) <>0 Then x[num,4]:=x[num,4]+2000;
        end;
      gotoxy(4,16);
      write('ORA DI ENTRATA: ');
      write(x[num,1],',',x[num,2],',',x[num,3]);
      gotoxy(4,18);
      write('ORA DI USCITA: ');
      write(h:',',m1:',',s);
      gotoxy(4,20);
      write('IMPORTO: ');
      write(x[num,4]);
      x[num,5]:=0; { cancellare macchina nella posizione i }
      cont:=cont-1;
      Delay(2000);
      gotoxy(4,16);
      write(' ');
      gotoxy(4,18);
      write(' ');
      gotoxy(4,20);

```

```

write('          ');
gotoxy(4,14);
write('VETTURE PRESENTI: ',cont:2);
If stampa Then
Begin
Writeln(LST,'Parcheggio Belleri Capozzi Ziliani & c. S.p.A. ');
Writeln(LST);
Writeln(LST,'ORA DI ENTRATA: ',x[num,1],':',x[num,2],':',x[num,3]);
Writeln(LST);
Writeln(LST,'ORA DI USCITA: ',h,':',m,':',s);
Writeln(LST);
Writeln(LST,'IMPORTO: ',x[num,4]);
Writeln(LST);
Writeln(LST,'Arrivederci e Grazie');
Writeln(LST);
End;
end
else
Begin
gotoxy(25,6);
write('PARCHEGGIO VUOTO',#7);
delay(2000);
End;
gotoxy(25,6);
write('          ');
gotoxy(45,18);
write('          ');
gotoxy(4,12);
write('          ');
end;

'A' : begin
gotoxy(45,18);
write('          ');
repeat
gotoxy(30,22);
write('Immetti il codice abbonato (max. 4 cifre) ');
num :=codice(true);
z := 0;
repeat
z := z+1;
until (num=abb[z]) Or (z >10) ;
gotoxy(45,24);
if z>10 Then Write('Codice Abbonato Errato ',#7);
until (z<=10);
gotoxy(45,24);
Write('ABBONATO RICONOSCIUTO',#7);
delay(2000);
gotoxy(30,22);
write('          ');
gotoxy(45,24);
write('          ');
end ;
'0' : Begin
gotoxy(45,18);
write('          ');
End;
else
begin
gotoxy(45,18);
write('SCELTA ERRATA');

```

```
    end;  
    end; { CASE }  
    UNTIL c='0';  
end.
```