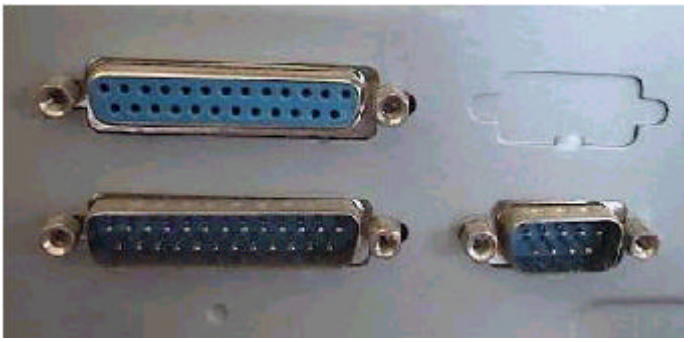
PORTA PARALLELA

La porta parallela del personal computer (detta anche LPT, porta stampante oppure centronics) è, tra le interfacce normalmente disponibili, certamente la più popolare presso gli hobbisti elettronici. Questa popolarità deriva dal fatto che la parallela presenta un discreto numero d'ingressi e uscite direttamente compatibili con gli usuali circuiti digitali e che la loro programmazione è piuttosto semplice, almeno in ambiente DOS.
In genere il comune PC possiede una sola porta parallela cui sono evidentemente connessi la stampante ed eventualmente altri dispositivi (ZIP, scanner…). Qualora s'intenda usare la LPT per circuiti autocostruiti, è cosa quanto mai opportuna installare almeno una seconda parallela. Questo evita da una parte la necessità di sconnettere la stampante ogni volta che si lavora e dall'altra, nel caso d'errori, non si rischia la rottura dei chip montati sulla scheda madre.
La porta parallela installata sul PC è facilmente riconoscibile: si tratta di un connettore a 25 poli DB25 femmina (cioè con i fori), posta normalmente sul retro del PC. Non va confusa con la porta seriale che, in vecchi PC, è costituita da un connettore DB25 maschio (cioè con gli "spilli"). Nella fotografia si vede il connettore della porta parallela (in alto) ed i connettori delle due porte seriali, un DB25 ed un DB9, attualmente più diffuso (in basso).



Un consiglio è di procurarvi un cavo di uno o due metri con doppio connettore DB25 maschio e i fili connessi uno ad uno (cioè il pin uno di un connettore connesso all'uno dell'altro connettore, il due al due e così via), al fine di poter comodamente lavorare senza spostarsi continuamente sul retro del PC. Molto utile anche un *gender* femmina-femmina e, eventualmente, il meno necessario maschio - maschio. Il connettore equivalente, in genere installato sulle stampanti, (centronics) è invece di 36 pin, pur avendo sostanzialmente la stessa funzione ed in genere è scomodo da utilizzarsi (troppi pin inutili…).
Nel tempo la porta parallela montata nel PC si è evoluta anche se, con qualche eccezione, si è mantenuta la compatibilità con i primi modelli.
Le categorie che comprendono tutte le porte attuali, sono sostanzialmente tre:

- La SPP – Standard Parallel Port. È il modello originario di porta parallela, installato sul primo PC XT. Pensato originariamente solo per la connessione alle stampanti di tipo centronics, con qualche accorgimento permette una certa flessibilità anche in altri usi. Tutti gli altri tipi di porta parallela sono compatibili con questa e quindi il software e l'hardware sviluppati per questa porta sono praticamente universali a condizione di seguire alcuni accorgimenti e non utilizzare alcune caratteristiche minori.

- La EPP – Enhanced Parallel Port. Permette lo scambio bidirezionale dei dati fornendo un supporto hardware per l'handshaking, pur permettendo praticamente tutte le funzionalità SPP. E' possibile raggiungere alte velocità di trasferimento, circa 10 volte maggiori della SPP.

- La ECP – Extended Capabilities Port. Permette una maggiore flessibilità d'utilizzo, ha il supporto del DMA ed una gran varietà di modi di funzionamento.

Questi modi di funzionamento sono regolati dallo standard IEEE 1284 del 1994, accanto al nibble mode ed al byte mode. In questo tutorial si tratterà solo della SPP, rimandando la trattazione degli altri modi ad altri documenti.
In seguito si farà riferimento, più che alle norme, al comportamento reale delle porte installate nei PC attuali e passati. Nel caso di comportamenti differenti o legati a situazioni particolari si evidenzierà la cosa al fine di mantenere la massima compatibilità anche a rischio di non ottimizzare le prestazioni, sconsigliando in particolare l'uso d'alcune tecniche che rischieranno di danneggiare il PC.
Prima di procedere alla descrizione delle porte, mi sembra opportuno indicare come configurare la porta nei vari modi di funzionamento, quando la cosa è ovviamente possibile.

- Se il PC è piuttosto vecchio e la porta parallela è installata su una scheda connessa al bus ISA (diciamo fino ad un 486 del 1994), molto probabilmente la porta è una SPP e quindi non necessita d'alcuna configurazione. In genere la stessa scheda che ospita il connettore stampante ha anche la funzione di controller per i dischi e le porte seriali .

- Se la porta è costituita da un chip installato sulla scheda madre in genere è configurabile per alcuni o tutti i modi previsti dalla norma IEEE 1284. Per far ciò è necessario, prima della fase di boot, accedere al menu di configurazione del setup (in genere premendo il tasto DEL o F1 durante il conteggio della memoria) e guardate alla voce *integrated peripherals* o simile. Per gli scopi che si vogliono raggiungere con questo tutorial la porta va configurata come *normal* oppure *centronics* oppure *default* oppure ancora *printer mode*, tutti termini sostanzialmente equivalenti. Se previsto, è anche possibile utilizzare il modo bidirectional. Se la porta è configurata come EPP, valgono tutte le cose che saranno dette, anche se non si sfruttano le peculiarità di tale modo di funzionamento. Prima di procedere, vi consiglio però di leggere l'avvertenza riguardante la sicurezza, riportata al termine del paragrafo.

- Se la porta è installata su una schedina aggiuntiva, è spesso disponibile un jumper che permette di configurare la porta come SPP o EPP.

Qui sotto, l'immagine della classica scheda di aggiuntiva, un investimento di cui non ci si pentirà. La scheda va installata in uno slot ISA libero, previa configurazione di alcuni jumper:

- LPT1, LPT2 o, in alcuni casi, LPT3. Ovviamente occorre evitare la prima opzione se è già presente la porta parallela principale
- IR5 o IRQ7. In genere è opportune non selezionare nessuno questi jumper perché, in PC normalmente configurati vi è penuria di interrupt e non sono normalmente necessari.
- Modo SPP o EPP, quando disponibile. Il modo EPP è più generale e quindi vi consiglio di attivarlo anche nel caso in cui voleste realizzare circuiti basati sulle idee presentate in questo tutorial.

Come già detto i PC hanno normalmente una sola porta parallela, indicata come LPT1 oppure PRN. Lo standard permette però di estenderne il numero fino a tre (su alcune macchine, soprattutto vecchie, fino a quattro). Le

porte aggiuntive sono indicate come LPT2 e LPT3.
Ciascuna porta parallela è indirizzabile attraverso una serie di indirizzi nello spazio di I/O, occupando un minimo di quattro indirizzi consecutivi; il primo di questi indirizzi è chiamato *indirizzo di base*. Il genere LPT1 ha indirizzo di base 0x378 (usando la sintassi C, con 0x378 s'intende il numero esadecimale 387, spesso indicato anche con 378h), LPT2 ha indirizzo 0x278 e LPT3 0x3BC. Gli indirizzi di tali porte non sono però fissi ma dipendono dall'hardware installato e da come il BIOS effettua la ricerca, causando spesso problemi nell'interpretazione corretta delle informazioni (per esempio la porta con indirizzo 0x3BC è indicato sulle vecchie macchine come LPT1…)

> Nei PC moderni la porta parallela è in genere gestita da un chip che controlla anche altre interfacce del PC (seriali, dischi). In caso di errori nell'uso dell'interfaccia si possono causare guasti che si ripercuotono sul funzionamento di tutto il PC, fino a causarne il blocco. Il chip non è in genere sostituibile e ciò causa la necessità di dover cambiare tutta la scheda madre, con costi e seccature facilmente immaginabili. Per questo vi consiglio vivamente di comprare una schedina parallela indipendente da installare su uno slot ISA libero (anzi compratene due, perché in genere si rompono l'ultima domenica prima delle ferie estive…). Se possibile prendetene una che supporti anche la modo EPP, certamente più flessibile e dallo stesso costo, circa 25.000 lire. L'unica difficoltà potrebbe essere quella di reperirla nel negozio sotto casa.

Due sono i modi per conoscere gli indirizzi effettivi:

- durante la fase di boot del PC, avere l'occhio attento e leggere sulla prima schermata gli indirizzi delle porte parallele che il BIOS individua, almeno nei PC che mostrano quest'informazione (è possibile bloccare il boot con il tasto Pause e leggere più facilmente il dato cercato).
- leggere il contenuto delle locazioni di memoria 0x408 e seguenti ed interpretare secondo la tabella seguente

0000: 0408 Indirizzo di base di LPT1
0000: 040A Indirizzo di base di LPT2
0000: 040C Indirizzo di base di LPT3
0000: 040E Indirizzo di base di LPT4

Gli indirizzi di memoria sono in esadecimale, nella sintassi *segmento,offset*.
Di seguito un frammento di codice che legge l'indirizzo di una porta in Turbo C in ambiente DOS.

```
unsigned int porta_base, porta_scelta;

porta_base = peek (0x40,0x8+2* (porta_scelta-1) )

printf("L'indirizzo della porta è 0x%X\ n", porta_base );
//porta_scelta assumere il valore 1 per la prima porta parallela e 2 per la seconda.
```

I PIN DELLA PORTA PARALLELA

Complessivamente sono disponibili sul connettore della porta parallela 12 bit per l'output e 5 per l'input.
Gli ingressi e le uscite sono TTL compatibili (alcuni possono essere a collettore aperto, in genere con resistore di pull-up integrato all'interno del PC) anche se le tensioni e le correnti effettivamente disponibili sono piuttosto variabili in funzione delle tecnologie impiegate per la costruzione della porta, in genere CMOS. Orientativamente un uno logico appare come una tensione molto vicina ai 5V, uno zero come 0V (a vuoto). Le correnti disponibili sono in genere di almeno 5 mA sia di sink sia di source ma spesso molto di più (anche 20 mA o più). Fanno eccezione i portatili e alcune vecchie schede: i primi possono erogare correnti piccole, le seconde hanno le uscite tipiche TTL-LS (Voh = 2.4V, Isource << Isink).

La seguente tabella riporta l'assegnazione dei pin sul connettore DB25

1	STROBE	Pin d'uscita di handshake
2...9	D0...D7	Pin per l'uscita di un byte di dati. La massima corrente che può erogata è di 26 mA nello stato alto e di 24 in quello basso
10	ACKNOWLEDGE	Pin d'ingresso. Insieme con il pin 1 permette di eseguire l'handshake
11	BUSY	Pin d'ingresso. Indica al calcolatore se la stampante è occupata
12	PAPER END	Pin d'ingresso. Indica se la carta è esaurita
13	SELECT	Pin d'ingresso. Indica che la stampante è attiva e pronta a ricevere i dati
14	AUTOFEED	Pin d'uscita. Indica alla stampante il modo d'avanzamento della carta
15	ERROR PRINT	Pin d'ingresso. Indica che la stampante non funziona correttamente
16	INIZIALIZE	Pin d'uscita. E' usato per reinizializzare la stampante
17	SELECT INPUT	Pin d'uscita. Abilita la stampante a ricevere i dati
18...25	GND	Linee di massa

I singolo bit dei tre registri hanno una precisa corrispondenza con i piedini del connettore

D7 9	D6 8	D5 7	D4 6	D3 5	D2 4	D1 3	D0 2	BASE
BUSY 11	ACK[1] 10	PE 12	SLCT 13	ERROR 15	***	***	***	BASE1
***	***	***	INT[2]	SLCINT 17	INIT 16	AUTOFD 14	STROBE 1	BASE2

[1] Il bit D6 di BASE 1, corrispondente al piedino 10 del connettore, è utilizzato per la generazione delle interruzioni.

[2] Il bit D4 di BASE 2, indicato con INT, è utilizzato per l'abilitazione delle interruzioni. (Il valore binario 1 abilita le interruzioni mentre lo 0 le disabilita).

I pin 1, 11, 14, 17 sono letti, o scritti, con un valore invertito, quindi per ottenere il valore reale é necessario negarli.

◊ I pin numerati con i numeri 1, 2, 3, 4, 5, 6, 7, 8, 9, 14, 16 e 17 sono in uscita dal calcolatore

◊ I pin numerati con i numeri 10, 11, 12, 13 e 15 sono in ingresso al calcolatore

◊ I pin da 18 a 25 sono collegati a massa.

Dal punto di vista software la porta parallela, utilizzando le istruzioni d'ingresso – uscita di qualsiasi linguaggio di programmazione, è vista come un insieme di tre registri.
Gli indirizzi dei registri sono:

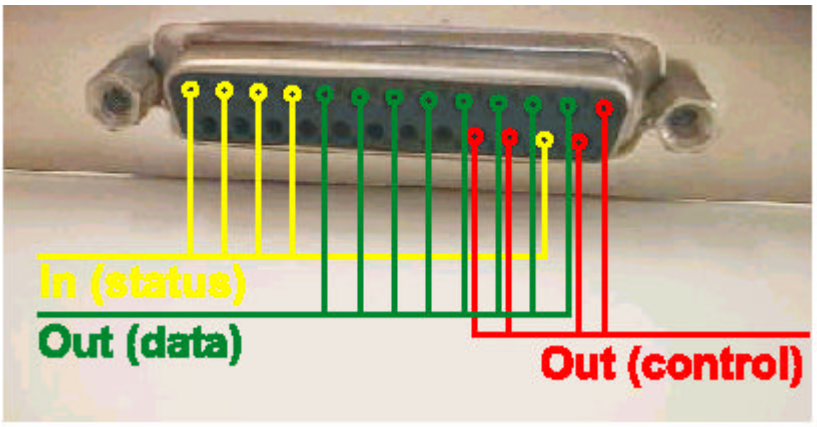
◊ BASE é l'indirizzo del registro utilizzato per l'uscita dei dati dal calcolatore verso la stampante

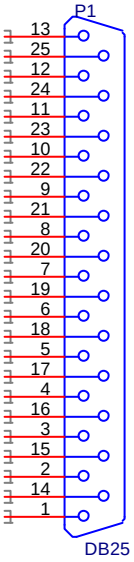
◊ BASE1 é l'indirizzo del registro che contiene lo stato della stampante

◊ BASE2 é l'indirizzo del registro utilizzato per l'invio dei dati di controllo verso la stampante

I bit che compongono BASE 1 sono in ingresso al calcolatore, mentre i bit di BASE e BASE 2 sono in uscita.

Realizzando un circuito connesso alla porta parallela è opportuno collegare tutti i fili di massa, sia per evitare il rischio di avere pin non connessi sia per diminuire gli eventuali disturbi.





L'immagine evidenzia i pin di uscita con l'indicazione dei relativi registri, visti dal connettore del PC. I pin non indicati sono tutti connessi a massa.
Verificate sempre con attenzione la corrispondenza dei pin se si usano prolunghe o simili, leggendo sul connettore stesso la numerazione e confrontando con la tabella precedentemente riportata.

Attenzione in particolare al fatto che, utilizzando un adattatore maschio la corrispondenza dei pin è, ovviamente, simmetrica, dato che deve incastrarsi in un connettore femmina.

I REGISTRI DI BASE

Ciascuna porta parallela configurata come SPP è controllabile, come affermato in precedenza, attraverso tre registri consecutivi nello spazio di I/O del processore, chiamati *data register (*BASE*), status register(*BASE1*) e control register(*BASE2*)*, ciascuno di 8 bit.
Per scrivere un byte in questi registri è necessario eseguire un'istruzione di *output* verso l'indirizzo interessato. Per leggere un byte è necessario effettuare un'istruzione di *input* Non è possibile leggere o scrivere un solo bit alla volta, indipendentemente dal linguaggio adottato (si tratta, infatti, di locazione di I/O, non di memoria).
Usando il Borland C (alla fine degli appunti sono riportate le istruzioni in Turbo C necessarie per l'interfaccimento egli operatori logici necessari per operare sui singoli bit di una variabile intera) la sintassi è la seguente (tale sintassi sarà adottata nel seguito del tutorial):

```
outportb (addr, dato);
dato = inportb (addr);
```

dove *dato* e *addr* sono rispettivamente il byte da leggere o da scrivere (variabile a otto bit, senza segno) e l'indirizzo del registro (variabile o costante a 16 bit, senza segno).
Usando la sintassi assembler:

```
MOV DX, ADDR
MOV AL, DATO
OUT DX, AL
```

```
MOV DX, ADDR
IN AL, DX
MOV DATO, AL
```

Una precisazione: quanto detto vale solo in ambiente MS-DOS. Come ben sanno i programmatori in ambiente Windows non è, infatti, possibile effettuare istruzioni dirette di input o output da parte di programmi utente ed in genere i linguaggi per tale ambiente neppure possiedono istruzioni o funzioni di input ed output (ma il problema può essere aggirato utilizzando delle routine in linguaggio assembler).

IL DATA REGISTER

Il data register è un registro di sola scrittura direttamente connesso agli otto pin di dato presenti sul connettore esterno della parallela. L'indirizzo è quello di base della porta (BASE).
Per impostare un pin alto o basso basta scrivere nel bit corrispondente di questo registro 1 oppure 0, rispettivamente. Il bit meno significativo del registro corrisponde al pin Dato0.
Se per esempio si vogliono impostare i pin in modo tale che i pin 2 (bit 0) e 7 (bit 5) siano alti ed i pin 3, 4, 5, 6, 8 e9 bassi, occorre eseguire la seguente istruzione (la costante DATA indica l'indirizzo del registro dei dati della porta interessata, per esempio 0x278):

```
outportb (DATA, 0x21);          // 0x21 = 00100001 in binario
```

Occorre infine notare che una volta scritto il byte nel registro, i pin di uscita non cambiano più il loro valore fino alla scrittura successiva perché il circuito di uscita della porta è formato da otto flip-flop.

LO STATUS REGISTER

Questo registro è di sola lettura ed ha indirizzo BASE 1. Quando il processore effettua la lettura di questo registro sono riportati i valori dei cinque pin in ingresso, secondo la tabella di seguito riportata. L'ultima colonna indica se è presente una porta not all'ingresso della porta parallela: nel caso ci sia un SI, vuol ricordare che un livello logico alto è letto come 0 logico. Se invece si trova scritto NO, un livello logico alto è letto come 1 logico.
Se voglio conoscere il valore dei pin 11 e 12 devo eseguire il seguente codice

```
data = inportb (STATUS);      // Leggo gli 8 bit di stato
data = data ^ 0x80;           // Inverto il bit più significativo con uno xor
busy = (data & 0x80) >> 7;    // Estraggo il pin 11 (bit 7)
po = (data & 0x20) >> 5;      // Estraggo il pin 12 (bit 5)
```

Il bit IRQ è resettato in caso d'interrupt attivato dalla linea Ack. Vale invece 1 qualora non si sia attivata nessuna interrupt.

IL CONTROL REGISTER

Questo registro è primariamente un registro di scrittura anche se, in alcuni casi, è possibile l'utilizzo in lettura.L'indirizzo è BASE 2.
Nel caso dei tre bit invertiti, la scrittura di un 1 causa sull'uscita una tensione di 0V. Per esempio per portare tutti i quattro pin della LPT2 ad un livello logico alto devono eseguire la seguente istruzione

```
outportb (CONTROL, 0x04);          // 0x04 = 00000100
```

Il bit 5 del registro di controllo permette di attivare la modalità bidirezionale delle porte che ne sono provviste: ponendo a 1 questo bit è possibile leggere i dati presenti sugli otto pin 2...9, attraverso la lettura del registro dei dati, come di seguito mostrato.

```
outportb (CONTROL, 0x10);          // 0x10 = 0001000 setto per l'ingresso
data = inportb (DATA);
```

Se il bit vale 0, è possibile scrivere nel registro dati, come in precedenza descritto.
Non tutte le porte hanno questa possibilità ed alcune funzionano con modalità diverse. Dalla mia esperienza posso affermare che le tutte le porte configurate come EPP hanno questo funzionamento. Le porte del PC XT originale, ed in genere le vecchie macchine, non hanno questo tipo di funzionamento.
Il bit 4 permette di abilitare le interrupt alla ricezione di un fronte sul pin Ack. Se l'interrupt è abilitato e la stampante connessa ad una linea del controllore di interrupt (p.e. attraverso l'apposito ponticello presente su molte schede), una transizione sul pin Ack causa un'interrupt. Il fronte attivo può cambiare da scheda a scheda. L'uso di tale possibilità richiede un'approfondita conoscenza delle problematiche correlate alla gestione delle interruzioni.

ISTRUZIONI INGRESSO USCITA IN C

USCITA DI UN BYTE IN LINGUAGGIO C

Si utilizza un compilatore Borland. Per avere un maggiore controllo è possibile utilizzare le istruzioni Assembler **in** ed **out,** utilizzando l'istruzione **asm**.

outport outportb

È necessario includere <DOS.H>

outport permette l'uscita di una word (16 bit) da una porta hardware

outportb permette l'uscita di un byte (8 bit) da una porta hardware

Sintassi

outport (unsigned Porta, unsigned Valore);

outportb (unsigned Porta, unsigned char Valore);

Porta é l'indirizzo della porta hardware d'uscita, ed é compreso tra 0 e 65535

Valore é il dato inviato in uscita. Non c'è nessun dato di ritorno

outport invia il byte basso all'indirizzo Porta ed il byte alto all'indirizzo Porta + 1.

outportb invia un byte all'indirizzo Porta.

Esempio:

outportb (888, 'A') // invia il carattere A alla stampante

888 é l'indirizzo Porta 'A' é il valore inviato in uscita

INGRESSO DI UN BYTE IN LINGUAGGIO C

inport, **inportb**

È necessario includere <DOS.H>

inport legge una word dalla porta hardware

inportb legge un byte dalla porta hardware

Sintassi :

unsigned inport (unsigned Porta);

unsigned char inportb (unsigned Porta);

Porta é l'indirizzo della porta sulla quale si preleva il dato ed é compreso tra 0 e 65535.

Il valore di ritorno é il dato letto.

inport legge il byte basso all'indirizzo Porta ed il byte alto all'indirizzo Porta+1

inportb legge un byte all'indirizzo Porta

Esempio:

unsigned valoreLetto;

unsigned Porta=888;

valoreLetto = inport (Porta)

OPERAZIONI TRA BIT

ingresso & uscita	AND bit a bit tra i due interi 011 & 101 ->001
ingresso | uscita	OR bit a bit tra i due interi 011 & 001 ->011
~ ingresso	NOT complemento di ingresso. 011->100
valore >> n	**shift a destra; muove i bit di Valore a destra di n, i bit sono persi ed a sinistra sono aggiunti degli zeri. 10001>>2 ->00100**
valore << n	shift sinistra; muove i bit di Valore a sinistra di n, i bit sono persi ed a destra sono aggiunti degli zeri. 10001<<2 ->10000

valore è un intero senza segno e n è un intero.

TRASFORMAZIONE DI UN NUMERO DECIMALE IN BINARIO

Assegnato un intero unsigned stampare sullo schermo la sua composizione binaria

```
#include <stdio.h>
void main () {
unsigned int a=85;
unsigned int x;
for (int i=0;i<=7;++i) {
        x = a>>i;
        x=x&1;
        printf ("%d  %d  \n", i, x);
 }
}
```

Esercizi

Ingresso ed uscita di bit

Inviare in uscita quattro bit dalla porta BASE2 e rileggerli attraverso i bit 11, 10, 13, 15 della porta BASE1

CIRCUITI APPLICATIVI

Presento ora qualche piccolo circuito applicativo utile per fare le prime esperienze con la porta parallela.Ho già detto del rischio di danni al PC e quindi vi invito nuovamente prestare la massima attenzione alla realizzazione del circuito e ad ***evitare di effettuare modifiche al circuito con il PC acceso.***

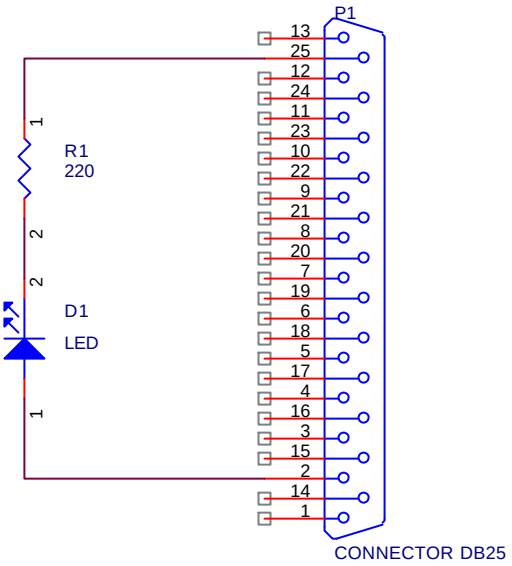
Intermittenza di un LED

Utilizzando la porta parallela si vuole progettare un circuito in grado di fare lampeggiare un LED.

Il circuito utilizzato per inviare un bit dal calcolatore verso esterno é abbastanza semplice.

Utilizziamo il bit D0 del registro d'uscita dei dati.

Nel caso in uscita sia necessaria una corrente superiore a quella fornita dalla porta parallela, è necessario inserire un buffer tra il connettore ed il LED o, più in generale, tra il connettore ed il circuito esterno.



Con il seguente programma si ottiene il lampeggiamento del LED

```
#include <dos.h>
void main(){
        unsigned dato=1;
        unsigned indirizzo;
        indirizzo = peek(0x40,08);          // lettura dell'indirizzo
        for(;;){                            // ciclo infinito
                outportb(indirizzo, dato);  // uscita del dato
                delay(1000);                // inserimento di un ritardo
                outportb(indirizzo, dato-1); // uscita del dato
                delay(1000);                // inserimento di un ritardo

        }
}
```

Utilizzando i bit della porta di controllo, si utilizza lo stesso procedimento.

ACQUISIZIONE DI UN BIT DALLA PORTA PARALLELA

Per acquisire un bit con la porta parallela é necessario utilizzare il byte di stato che ha l'indirizzo BASE1.

Utilizziamo il bit D7, corrispondente al piedino 11 del connettore della porta parallela. Il bit D7 è acquisito in modo negato ed é necessario ripristinare il suo valore originale.

I passi da compiere, per acquisire il bit sono:

1. ricavare l'indirizzo base della porta
2. lettura del byte in BASE + 1
3. ricavare il bit D7 dal byte letto nel punto 2
4. negare il bit del punto 3

Il programma é il seguente

```
#include <dos.h>
void main() {
    int indirizzo, BASE_1, lettura, bit;
    indirizzo  =  peek(0x40,08);          // lettura dell'indirizzo
    BASE_1=indirizzo+1;              // calcolo dell'indirizzo di stato
    lettura = inportb(BASE_1);        // lettura del byte dalla porta
    bit(lettura >> 7 ) & 1;          // si ricava il bit D7
    bit = (~bit )&1;                          // si ricava il bit originale
    /* programma per l'utilizzazione del valore letto */
    .....
}
```

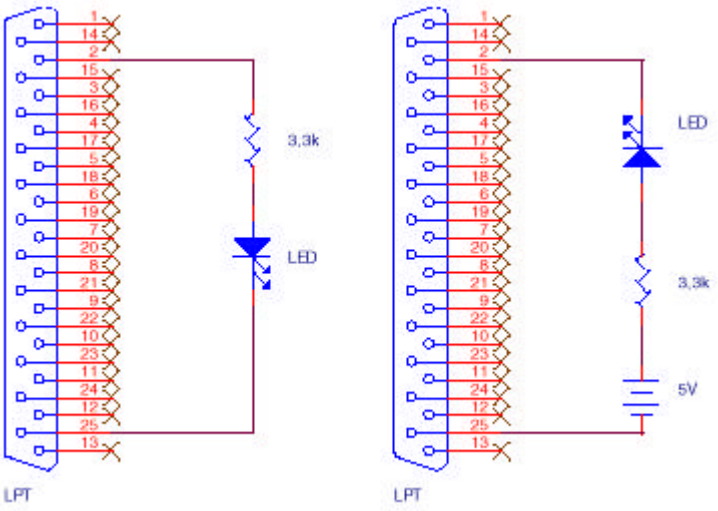
ACCENSIONE DI UN LED

Un paio di circuiti veramente semplici per accendere un led permettono di vedere come usare la parallela. È necessario disporre di una resistenza di circa 3,3k ed ovviamente di un led e, limitatamente al secondo circuito, di una batteria o un alimentatore da 5 volt (o anche 4,5V).
Il primo schema non necessita di particolari spiegazioni: quando il pin 2 è posto alto, il led si accende, quando è posto basso, si spegne.
Il codice relativo è banale.

```
outportb (DATA, 0x01); // accendo il led collegato al pin 2 (D0)
Ritardo;
```

outportb (DATA, 0x00); // spengo tutti i LED

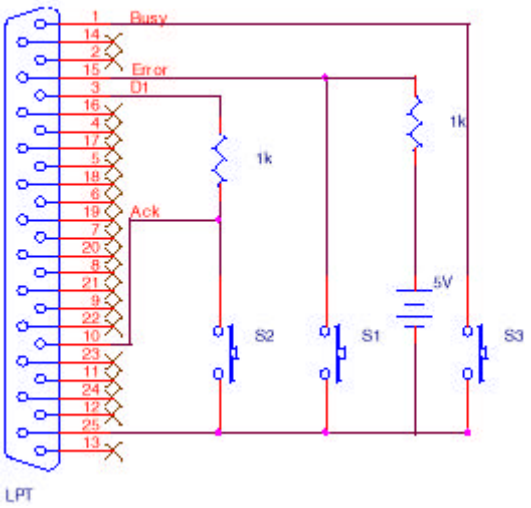
Occorre ricordare che, una volta acceso il led, esso rimane tale fino allo spegnimento del PC o finché è esplicitamente spento, indipendentemente dal fatto che il programma sia o no in esecuzione, in quanto memorizzato dall'hardware.



La connessione utilizzata in questo primo circuito (il led è acceso dalla corrente uscente dal pin della porta parallela) ha il vantaggio di non richiedere nessuna alimentazione esterna ma il difetto che su alcune porte parallele (in particolare vecchie o di PC portatili) potrebbe non funzionare a causa del fatto che la corrente di source è troppo piccola.
Il secondo circuito utilizza una sorgente di alimentazione esterna e dovrebbe funzionare con tutte le porte parallele.
In questo caso è lo **zero** che accende il LED. Per aumentare la luminosità del led è possibile diminuire il valore della resistenza anche a valori più piccoli, quale 330 ohm.
Ovviamente i due circuiti possono essere ampliati connettendo altri sette led ai pin di dato.

L'USO DI INTERRUTTORI

Nello schema seguente sono illustrati tre modi per connettere un interruttore alla porta parallela. In tutte e tre i casi l'interruttore chiuso è letto come tensione di 0 volt, l'interruttore aperto come tensione di circa 5 volt. Lo schema utilizzato con S1 richiede la presenza di una batteria esterna da 5V. Per conoscere lo stato dell'interruttore è sufficiente leggere lo stato del bit Error nel registro di controllo (oppure di un altro bit dello stesso registro se connesso ad un altro pin). È possibile la connessione con uno qualunque dei bit gestiti attraverso il registro di controllo. In certi casi questo circuito funziona anche senza batteria e senza resistenza ma l'estrema semplicità si paga con errori di lettura occasionali, soprattutto in ambiente rumoroso. Lo schema usato per S2 è simile al precedente, con la differenza che la tensione non è prelevata da una batteria ma da uno dei pin della porta dati, posto preventivamente a uno logico. Lo svantaggio è di richiedere per ciascun interruttore due pin della porta parallela, oltre la massa.

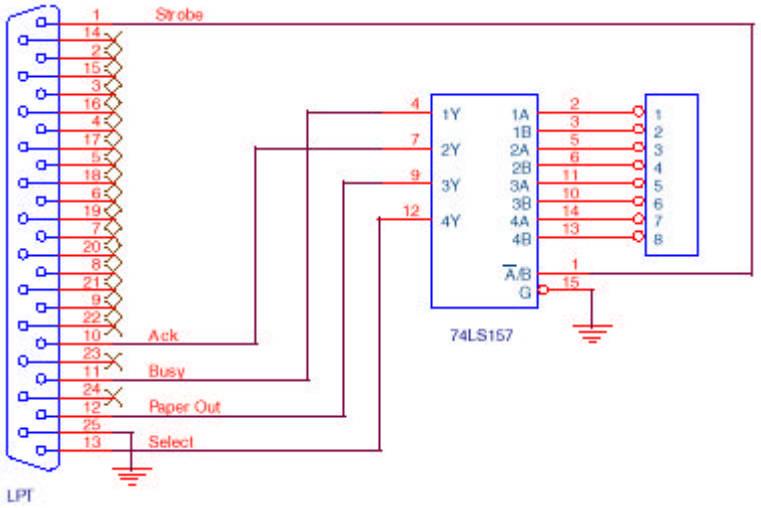


Lo schema utilizzato per S3 è il più semplice e funziona con i pin associati al registro di controllo; non richiede nessun componente oltre l'interruttore ma funziona esclusivamente a condizione che il pin cui è connesso l'interruttore sia a collettore aperto e che la resistenza di pull-up sia montata internamente alla scheda. In questo caso, per leggere lo stato dell'interruttore occorre impostare a 1 il pin scrivendo nel registro di controllo e poi leggere il bit nello stesso registro.

ATTENZIONE: qualora la porta non sia a collettore aperto (come succede in molte parallele moderne) nel caso della lettura su S3 si rischia la distruzione della porta. Pertanto si sconsiglia l'uso di questo metodo

LETTURA DI 8 BIT

Il seguente schema permette di leggere otto bit anche da porte non bidirezionali o senza uscite open collector. E' utilizzato un multiplexer comandato dall'uscita di strobe, leggendo quattro bit alla volta.



Il difetto di tale metodo deriva dalla relativa lentezza (richiede due letture e due scritture della porta) e dal fatto che è ovviamente richiesta un'alimentazione esterna per l'integrato (non disegnata nello schema). Il vantaggio è che funziona con qualunque tipo di porta senza modifiche.
Il codice da eseguire, è il seguente:

```
outportb (CONTROL, 0x01);          // Seleziono il nibble basso
dataLow = inportb (STATUS) & 0xF0; // Leggo il nibble basso
dataLow = dataLow >> 4;            // Allineo i quattro bit
outportb (CONTROL, 0x00);          // Seleziono il nibble alto
dataHigh = inportb (STATUS) & 0xF0; // Leggo il nibble alto
data = dataHigh | dataLow;         // Unisco i due nibble
data = data ^ 0x88;                // Sistemo i due bit invertiti
```

Ovviamente se la porta è bidirezionale oppure se è possibile usare in ingresso i quattro bit associati al registro di controllo vi sono minori problemi, pagati con una minore compatibilità tra porte diverse. La soluzione più razionale nel caso in cui serve l'input di una quantità significativa di dati, è però quella di adottare una porta EPP, sicuramente più standardizzata e veloce.

Esercizi di programmazione

1. Assegnato un intero senza segno, trovare il valore del bit nella posizione D6
2. Scomporre un numero intero senza segno in binario, inserendo i bit in un vettore. I bit nel vettore debbono essere rappresentati a partire dal bit più significativo.
3. Assegnato un intero senza segno, scrivere un valore logico 1 nella posizione D7
4. Assegnato un intero senza segno negare il bit nella posizione D3

```
#include <iostream.h>
```

```
void main(){
int a=123,x;
//parte a
for(int i=0;i<=7;++i){
x=a>>i;          // shift del bit
x=x&1;           // AND con 1 logico per isolare il bit
cout <<x ;
}
cout <<endl;
//parte b

x=a>>5;              // shift del bit
x=x&1;               // AND con 1 logico per isolare il bit
cout << " bit D6  "  <<x << endl;
cout <<endl;
//parte c

x=1;
x=x<<7;
cout <<x << endl;
int r=a|x;
cout <<r << endl;

// controllo
cout << "controllo"<< endl;


for(int i=0;i<=7;++i){
x=r>>i;
x=x&1;     // AND con 1 logico per isolare il bit
cout <<x ;
}
cout <<endl;

 // parte d

int shift=7;
int valore;
valore=(a>>shift)&1; // ricavo il bit shift

cout << "Valore= "<< valore;
if(valore==0){
     x=a|255;     // se valoe 0 OR con 255
     cout << " valore con 0 originale "<<x << endl;
}
else
{
     x=1;                 // se vale 1 AND con uno nella posizione cercata
     x=x<<shift;
     x=x&a;
```

```
    cout <<  "valore con 1 originale "<<x << endl;
}


cout << "controllo"<< endl;
for(int i=0;i<=7;++i){
x=r>>i;
x=x&1;
cout <<x ;
}
cout <<endl;
}
```

ESERCIZI DI INTERFACCIAMENTO

1. Accensione di un LED nel piedino 7 della porta parallela
2. Lampeggio di un LED nel piedino 3 della porta parallela
3. Uscita di un bit dal piedino 8 della porta parallela e sua lettura nel piedino 10 della stessa uscita di 4 bit dalla porta parallela, loro lettura dalla stessa porta e visualizzazione del dato letto
4. Generare, e visualizzare su un oscilloscopio, i seguenti segnali:

T=10mS