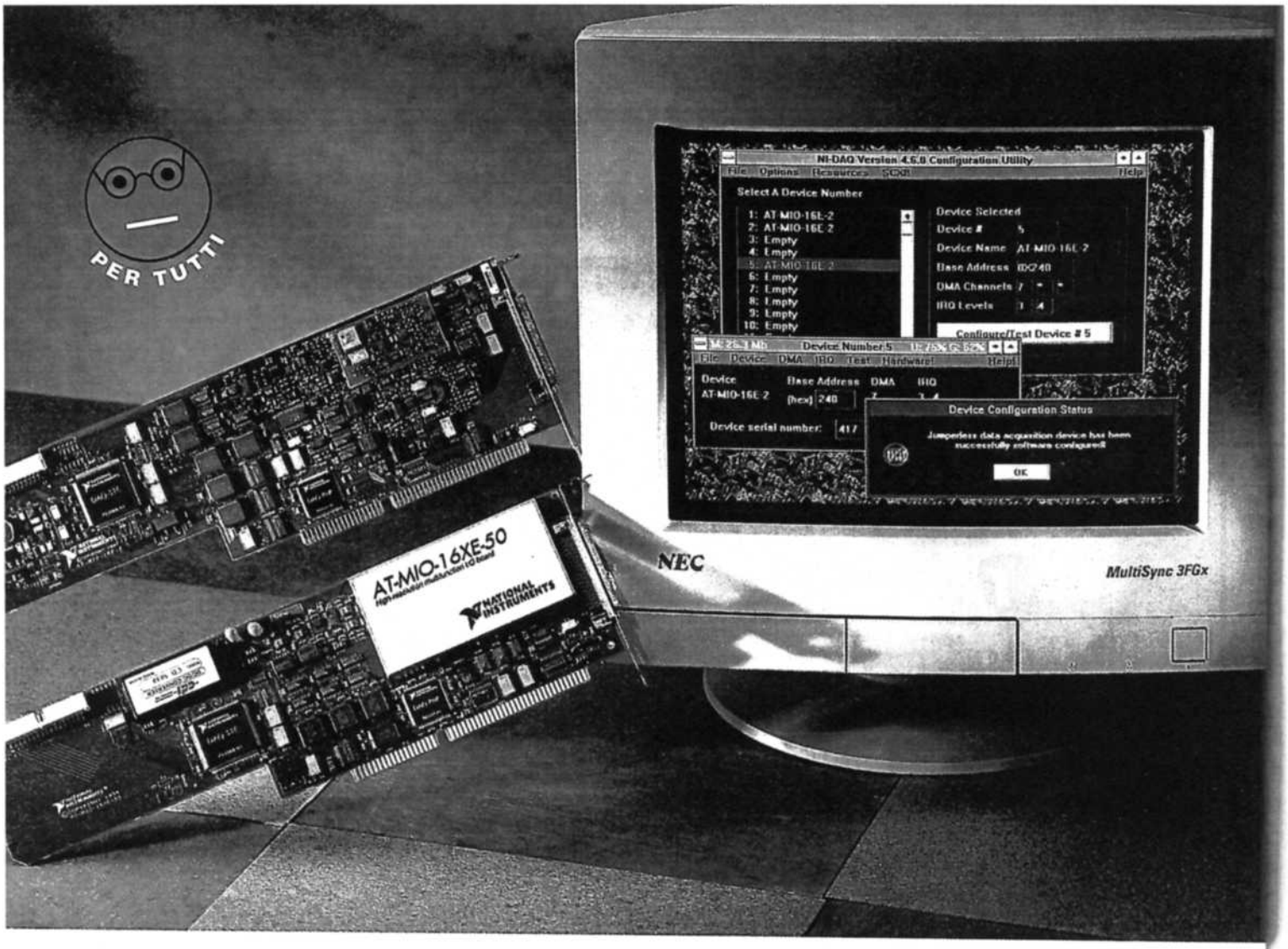


Comunicazione seriale asincrona

Quali sono gli standard e le problematiche relative ad una comunicazione seriale asincrona? Scopriamolo insieme

Prima parte

Giovanni Argentini

Ormai la maggior parte degli apparecchi commerciali dialoga per mezzo di una porta seriale.

Se prendiamo per esempio un banale multimetro dell'ultima generazione, vediamo che esiste la possibilità di interfacciarsi con un computer, come pure avviene nel caso di strumenti più costosi come frequenzimetri, oscilloscopi, analizzatori di vario genere ecc. Si potrebbe pensare che questo vale solo per le attrezzature dedicate in modo particolare

al mondo dell'elettronica pratica, ma così non è: chi di voi non ha mai visto un'agenda elettronica? Anch'essa dialoga con il computer attraverso tre soli fili.

E che dire dei telefoni cellulari, delle fotocopiatrici più recenti, dei fax, dei gruppi di continuità, dei modem?

Sono tutti apparecchi che nella maggioranza dei casi possiedono una linea seriale (anche se poi viene usata raramente) ed un software di dialogo per il computer.

Ma nonostante tutte queste applicazioni ancora oggi in pochi ne conoscono appieno il funzionamento.

La pratica vuole che si infili il connettore nella presa e che si attivi il software predisposto.

Ma se poi non si ha il funzionamento desiderato, come ci si deve comportare? La ricerca del guasto su una linea seriale non è banale come può sembrare a prima vista.

Per eseguire un esame piuttosto approfondito dei vari segnali l'unica soluzione è quella di monitorarli tutti ed otto con un analizzatore di stati logici, ma poiché questo strumento ha un costo tutt'oggi troppo elevato per gli hobbisti, si devono cercare strade meno costose.

Le tensioni in gioco

Il primo passo da affrontare nell'analisi di una comunicazione seriale consiste nell'esaminare la tensione dei diversi segnali.

Per garantire una discreta immunità ai rumori, qualcuno ha pensato di assegnare al livello logico "1" una tensione negativa ed al livello logico "0" una tensione positiva; sempre rispetto alla massa.

Quanto valgono queste due tensioni? Purtroppo lo standard richiede che tali tensioni non siano fissate, ma che possono variare in un range da +3 a +15 volt per la positiva e da -3 a -15 volt per la negativa.

Diciamo purtroppo perché anche gli schemi di interfaccia possono adattarsi ai circuiti non sempre in modo ottimale.

Ad esempio se qualcuno ha mai avuto dei computer portatili, si sarà certamente accorto che non di rado qualche apparecchiatura che dovrebbe dialogare con la porta seriale in realtà non la vede nemmeno.

I motivi di questo mutismo reciproco sono essenzialmente due: uno può essere un leggero scostamento dagli standard fissati, l'altro l'interfaccia dell'apparecchio esterno non compatibile con una seriale standard.

Nel primo caso è possibile, tanto per citare uno dei molteplici esempi, che la tensione della batteria del portatile sia scesa sotto una ben precisa soglia e che quindi le due tensioni sulla porta non siano più nell'intorno dei + o - 3 volt richiesti.

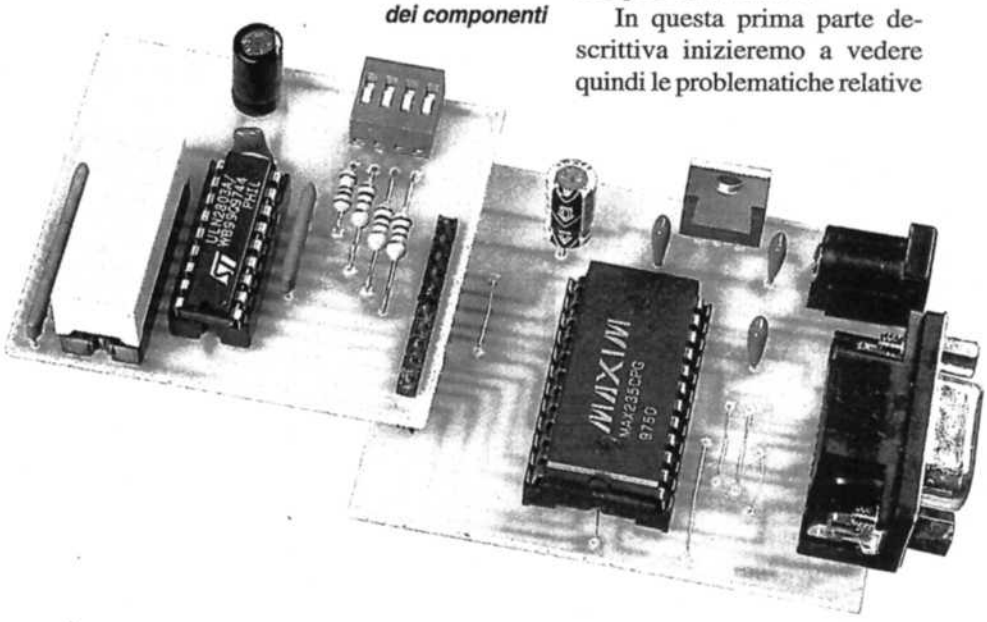
Nel secondo caso invece, poiché spesso si tende a risparmiare denaro fino all'inverosimile, capita che un apparato dichiarato compatibile con una porta standard RS232 in realtà non abbia un circuito d'interfaccia "regolare", ma magari implementato con due o tre diodi solamente per risparmiare il costo di un MAX232 o similari.

In questi casi non sarà mai possibile vedere tale apparato dialogare con il computer, perché è come se parlassimo molto piano e molto distante ad una persona che ha un udito debolissimo: ovviamente non saremmo ascoltati.

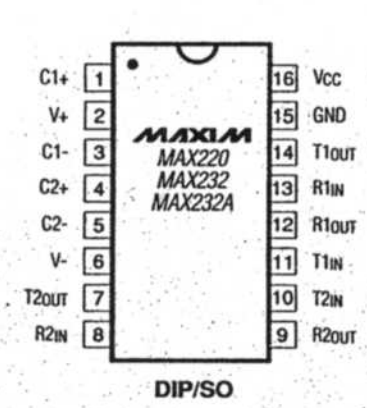
La prima condizione da rispettare quindi per una buona comunicazione seriale RS232 è il rispetto delle tensioni dei diversi segnali che, come abbiamo già anticipato, devono essere comprese tra +3 e +15 volt e tra -3 e -15 volt.

In realtà la maggior parte dei circuiti integrati lavora con tensioni di +5 e -5 volt per semplicità circuitale.

In questa prima parte descrittiva inizieremo a vedere quindi le problematiche relative



Ecco il risultato finale: ridotto ingombro e ordine nella disposizione dei componenti



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

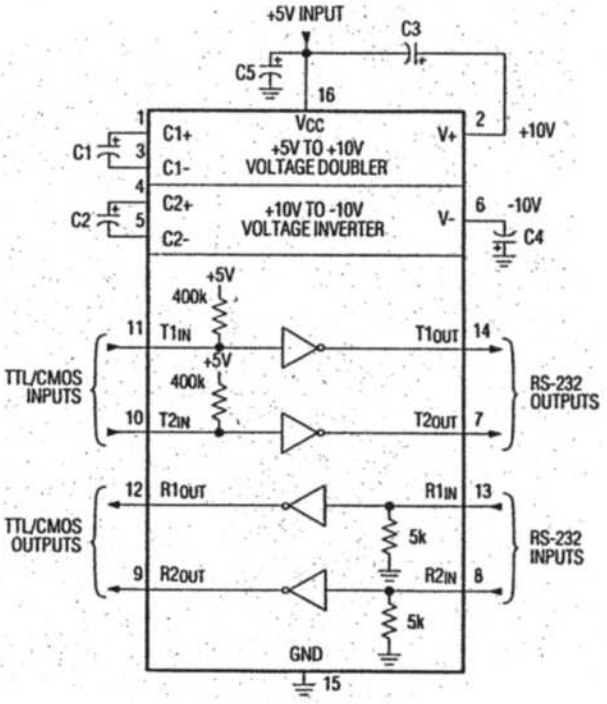
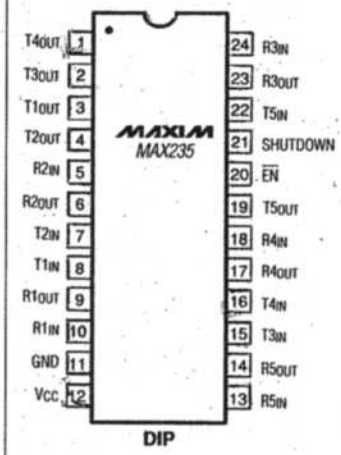


Figura 1. Piedinatura e diagramma a blocchi del MAX232

all'interfacciamento hardware, mentre nella seconda parte tratteremo il protocollo software, ovvero come far dialogare due persone con un linguaggio comune.

I convertitori di livello di linee seriali

Per adattare i livelli delle linee seriali standard ai livelli che comunemente vengono impiegati nei classici circuiti digitali, e cioè 0 volt per lo zero logico e 5 volt per l'uno logico, si trovano molti componenti sul mercato, diversi sia per caratteristiche che per dimensioni.



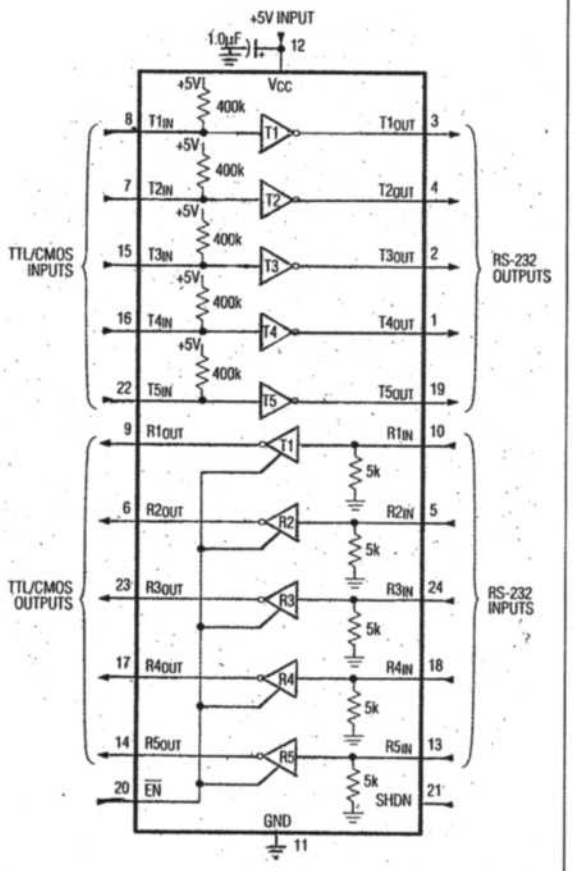


Figura 2. Piedinatura e diagramma a blocchi del MAX235

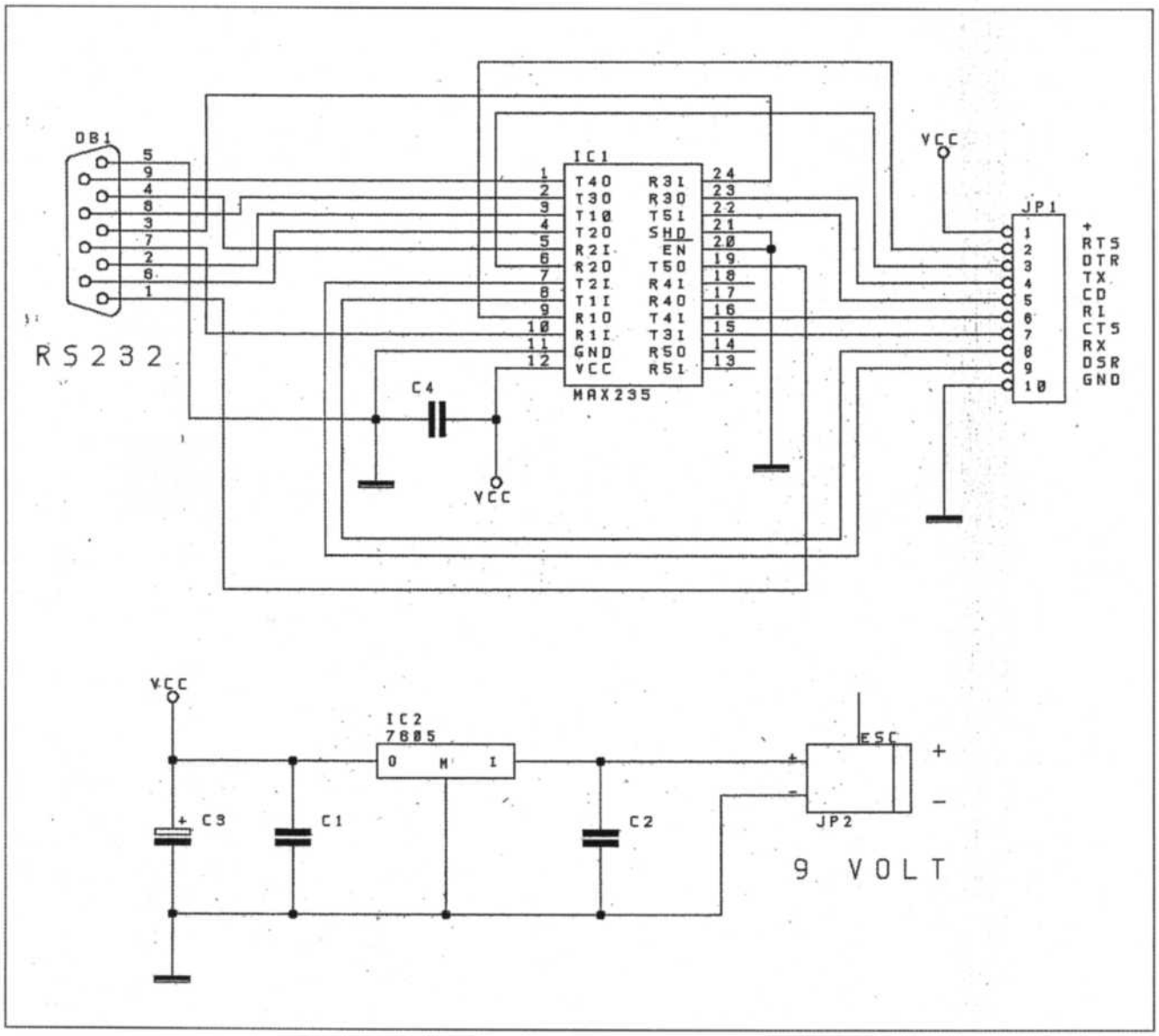


Figura 3. Schema elettrico del circuito di test

Il più conosciuto di tutti è il MAX232 della Maxim, azienda leader nella produzione di interfacce di qualsiasi tipo.

Questo integrato, il cui diagramma a blocchi è visibile in figura 1, ha al suo interno due blocchi funzionali che servono per la generazione di due tensioni simmetriche di 10 volt rispetto alla massa.

Il primo blocco prende in ingresso i 5 volt di alimentazione e li moltiplica per due ottenendo così i +10 volt per il ramo positivo. Il secondo blocco invece prende parte di questi +10 volt e li inverte producendo i -10 volt del ramo negativo.

Sono poi disponibili due buffer per la conversione da 232 a TTL ed altri due buffer per la funzione opposta.

Dicevamo che questo integrato è generalmente il più impiegato perché ha un costo relativamente basso e poi perché ha un numero di buffer sufficiente per la maggior parte delle applicazioni.

Perché il numero di buffer può essere importante? Non abbiamo ancora accennato al numero di fili che compongono una linea seriale, ma possiamo anticipare che sono in tutto 8.

La maggior parte di questi però non viene quasi mai impiegata e quindi essendo non significativo convertire segnali che non servono, si vanno a convertire solamente quelli che vengono impiegati, che in alcuni casi potrebbero anche essere uno soltanto, ovviamente esclusa la massa!

In effetti spesso accade che un'interfaccia sia di solo ingresso o di sola uscita, e quindi il segnale da convertire sarà solamente il TX o l'RX.

Se facciamo caso sempre alla figura 1, notiamo che i buffer provenienti dalla sezione TTL hanno delle resistenze di pull-up da 400 kΩ mentre quelli provenienti dalla sezione RS232 hanno delle resistenze di pull-down da 5 kΩ.

L'integrato MAX232 non è però il solo della MAXIM a convertire TTL/CMOS/-232, ma ne troviamo altri con caratteristiche e package diversi.

Caratteristiche elettriche del MAX235

Il MAX235

Uno di questi è il MAX235 (vedi figura 2), lo stesso che abbiamo impiegato per il nostro circuito di test.

Come si può subito vedere, il numero dei buffer cambia da 2 + 2 a 5 + 5, ovvero sarà possibile collegare fino a 5 uscite RS232 e fino a 5 ingressi RS232.

In questo modo, poiché le linee della seriale standard sono 8, sarà possibile convertirle tutte con lo stesso integrato.

La prima differenza che salta agli occhi rispetto al MAX232 è il fatto che i buffer collegati agli ingressi RS232 hanno sulla loro uscita un controllo trhee-state.

Questo implica che sarà possibile connettere sulla medesima porta seriale più MAX235 ma soltanto quello selezionato dal pin ENABLE entrerà in funzione.

Tabella 1

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 Input Threshold Low	T_A = +25°C, V_{CC} = 5 V	Norma Operation $\overline{SHDN}$ = 5 V (MAX223), SHDN = 0 V (MAX235-MAX241)	0.8	1.2		V
		Shutdown (MAX223) $\overline{SHDN}$ = 0 V, EN = 5 V (R4, R5)	0.6	1.5		
RS-232 Input Threshold High	T_A = +25°C, V_{CC} = 5 V	Normal Operation $\overline{SHDN}$ = 5 V (MAX223), SHDN = 0 V (MAX235-MAX241)		1.7	2.4	V
		Shutdown (MAX223) $\overline{SHDN}$ = 0 V, EN = 5 V (R4, R5)		1.5	2.4	
RS-232 Input Hysteresis	V_{CC} = 5 V; no hysteresis in shutdown		0.2	0.5	1.0	V
RS-232 Input Resistance	T_A = +25°C, V_{CC} = 5 V		3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 1.6 mA (MAX231-233 IOUT = 3.2 mA)				0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0 mA		3.5	V_{CC} - 0,4		V
TTL/CMOS Output Leakage Current	0 V ≤R_{OUT} ≤V_{CC}; EN = 0 V (MAX223); $\overline{EN}$ = V_{CC} (MAX235-241)			0.05	±10	µA
Receiver Output Enable Time	Normal operation	MAX223 MAX235-MAX241		600 400		ns ns
Receiver Output Disable Time	Normal operation	MAX223 MAX235-MAX241		900 250		ns ns
Propagation Delay	RS-232 IN to TTL/CMOS OUT, C_L = 150 pF	– tPHLS tPLHS		 4 6	0.5 40 40	10 µs
Transition Region Slew Rate	MAX223, MAX230, MAX234-MAX241 T_A = +25°C, V_{CC} = 5 V, R_L = 3 kΩ to 7 kΩ, C_L = 50 pF to 2500 pF, measured from + 3 V to -3 V or -3 V to +3 V		3	5.1	30	V/µs
	MAX231, MAX232, MAX233 T_A = +25°C, V_{CC} = 5 V, R_L = 3 kΩ to 7 kΩ, C_L = 50 pF to 2500 pF, measured from +3 V to -3 V or -3 V to +3 V		4	30		
Transmitter Output Resistance	V_{CC} = V+ = V- = 0 V, V_{OUT} = ±2 V		300			Ω
Receiver Out Short-Circuit Current	–			±10		mA

Una seconda differenza è il pin di SHUTDOWN: con questo ingresso è possibile "addormentare" il MAX235 (in pratica vengono spenti il convertitore ed il duplicatore di tensione) ottenendo così un notevole risparmio di corrente.

Anche il numero di pin ovviamente cambia: da 16 pin si passa a ben 24 pin ed il passo dell'integrato da 300 mils passa a 600. In tabella 1 ne vediamo le caratteristiche elettriche.

Quelle più importanti da considerare sono la resistenza di ingresso di 5 kΩ, i tempi legati all'abilitazione e disabilitazione dei buffer ricevitori che valgono rispettivamente 400 e 250 ns, il ritardo di propagazione che è di 4 us la resistenza di uscita che vale 300 Ohm ed infine la corrente assorbita che vale circa 10 mA.

Il nostro circuito

Per capire al meglio il funzionamento della porta seriale abbiamo realizzato un circuito di test il cui schema elettrico è visibile in figura 3.

Tutte le 8 linee interessate vengono gestite dal convertitore di livello IC1. Poiché quest'ultimo ha al suo interno ben 10 buffer, due di questi resteranno inutilizzati.

Il pin di ENABLE è stato mantenuto sempre attivo (basso livello logico) mentre il pin di SHUTDOWN è stato disattivato (livello logico basso) poiché non ci interessa il basso assorbimento a riposo in quanto l'alimentazione viene prelevata da una fonte esterna e regolata da IC2 un classico 7805.

Abbiamo scelto un 7805 e non il più piccolo 78L05 perché successivamente collegheremo a questo circuito un altro circuito di test che necessita di una certa corrente per l'accensione di alcuni led.

Le otto line convertite sono RTS, DTR, TX, CD, RI, CTS, RX e DSR. Tre di queste sono di uscita dalla porta, le altre sono di ingresso.

I buffer che non sono stati impiegati non sono stati collegati perché, come abbiamo visto, hanno già una resistenza interna al chip da 5 kΩ che li collega a massa.

Il circuito di test non presenta grosse difficoltà realizzative, ma è sicuramente consigliato l'impiego di un circuito stampato.

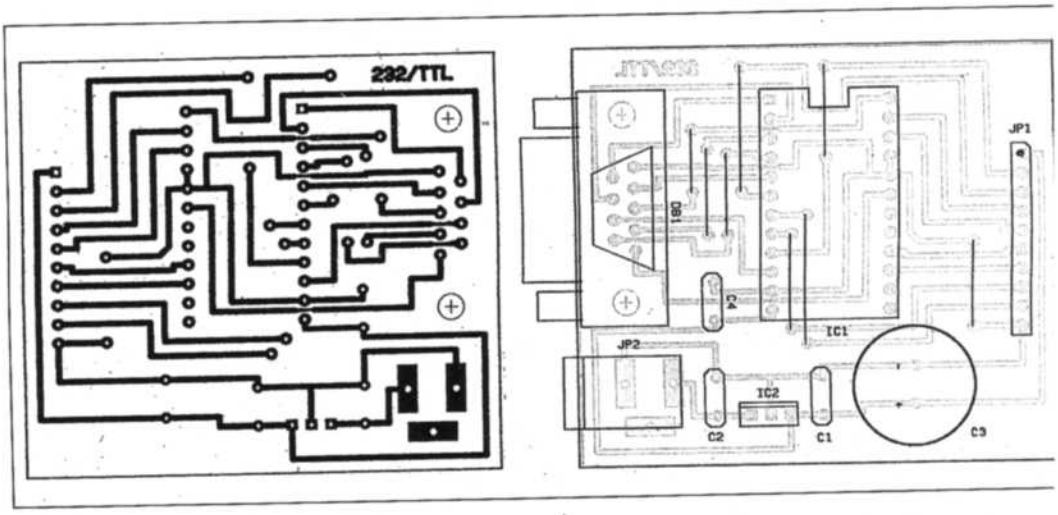


Figure 4 e 5 Circuito stampato, scala 1:1 e disposizione dei componenti

ELENCO COMPONENTI

Semiconduttori
IC1: MAX235
IC2: 7805

Condensatori
C1, C2, C4: 100 nF
C3: 47 µF

Varie
Presa DB9 90° da c.s.
Spina punto-linea
di alimentazione per c.s.

Montaggio

In figura 4 ne troviamo una possibile traccia.

Per non dover ricorrere ad un doppia-faccia che non risulterebbe di facile realizzazione per tutti, abbiamo deciso di approntare il tutto su di un solo lato ma con l'ausilio di ponticelli realizzati con di filo di rame.

In figura 5 si può vedere il piano di cablaggio dei componenti: i ponticelli da saldare sono otto, di cui tre sotto il circuito integrato IC1.

È indispensabile quindi, prima di montare lo zoccolo di tale integrato, ricordarsi di fissare i tre ponticelli sottostanti.

Per non rialzare troppo lo zoccolo, si consiglia di far uso di fili di piccolo diametro, anche senza guaina.

Si passerà poi al completamento dei ponticelli e successivamente si piazzeranno tutti gli altri componenti avendo l'accortezza di lasciare per ultimi i connettori.

La basetta è così finalmente completata.

Con la rivista del prossimo mese vedremo come impiegarla abbinandola ad un altro specifico circuito di test appositamente preparato. ■

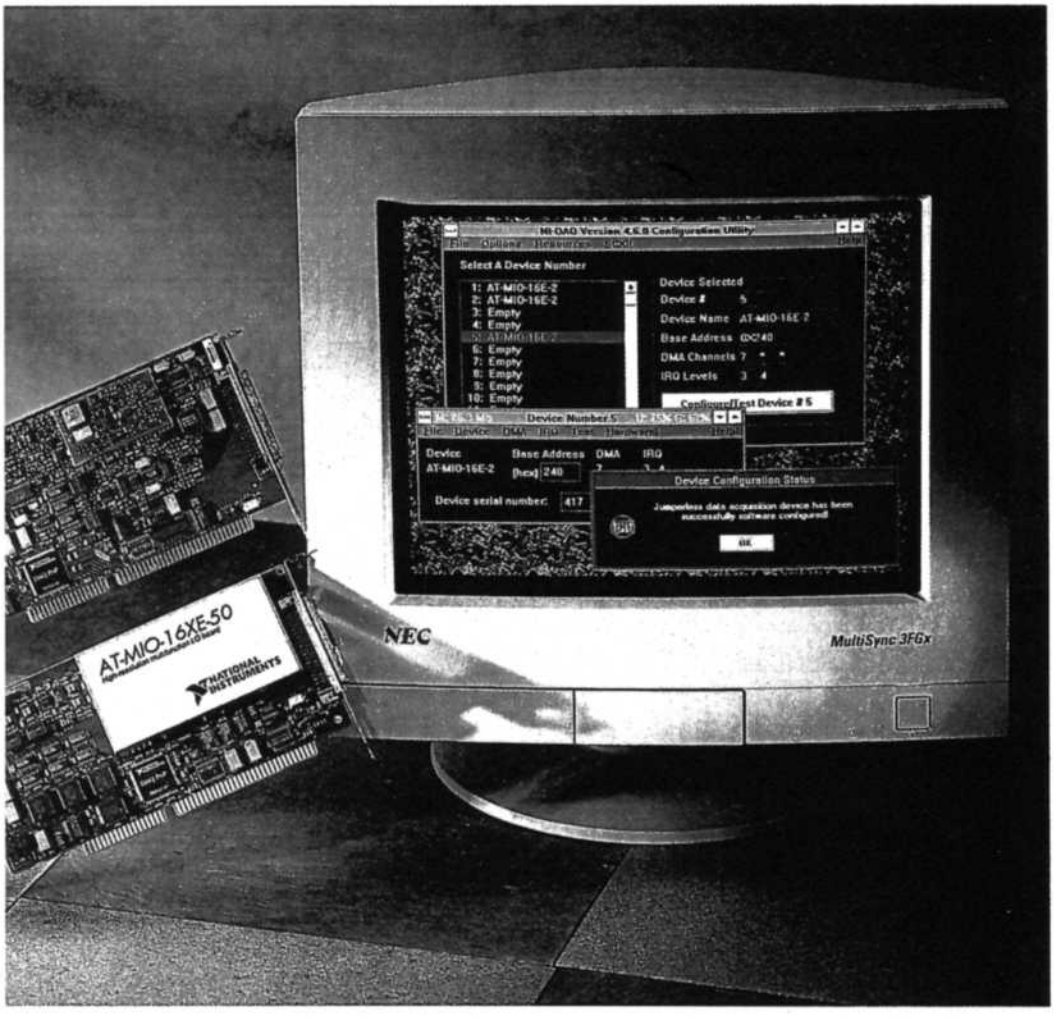
Comunicazione seriale asincrona

Quali sono gli standard e le problematiche relative ad una comunicazione seriale asincrona? Scopriamolo insieme

Seconda parte

Giovanni Argentini

CRITICO



Riprendiamo la trattazione della porta seriale intrapresa il mese scorso dedicandoci questa volta al lato software della questione.

Per far comprendere quanto sia importante il protocollo di dialogo tra due apparecchiature, facciamo subito un piccolo esempio. Supponiamo che due persone che non si conoscono si incontrino e che desiderino scambiarsi delle informazioni.

In questo caso la sezione "hardware" è identica, ovvero entrambi parleranno con la bocca ed ascolteranno con le orecchie. Ma se per ipotesi una persona parla la lingua italiana e l'altra la lingua inglese, lo scambio di informazioni non potrà avvenire.

Una soluzione al problema potrebbe essere lo studio da parte di una delle due persone della lingua dell'altro interlocutore, oppure l'adozione di un linguaggio più universale come quello delle gesta.

In ogni caso, affinché ci sia un dialogo tra i due, è fondamentale che ci siano delle regole conosciute ad entrambi. Lo stesso dicasi per far dialogare due apparecchiature sulla porta seriale: la prima cosa che entrambe devono conoscere è la "lingua" comune con cui parlarsi.

Lo standard asincrono

Non abbiamo volutamente detto RS232 poiché come abbiamo già accennato il mese scorso lo standard RS232 è definito sì dal protocollo software, ma soprattutto da quello hardware.

Non è escluso infatti che lo stesso protocollo venga impiegato anche per lo standard RS485 o RS422, differenti solo per l'hardware usato.

Una comunicazione si dice asincrona quando tra due interlocutori non esiste un segnale di clock che sincronizzi ogni dato, come per esempio accade per il bus IIC o il Microwire o l'SPI.

Ciò significa che l'interlocutore che inizia a parlare deve rispettare alcune regole affinché l'altro interlocutore possa capire ciò che gli viene inviato.

Nello standard che ci interessa vedere, il dato viene inviato con un insieme di dati detto "pacchetto" o "treno di impulsi".

Il primo bit che viene spedito è il bit di start.

Questo bit indica al ricevente che sta per giungere un insieme di altri bit.

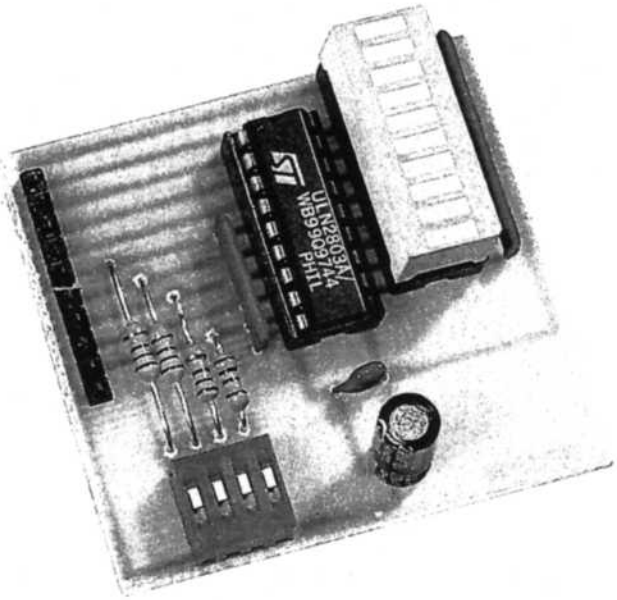
Quanti? Esattamente quanti stabiliti in precedenza con le regole di conversazione tra i due. La durata dello start bit è fondamentale, lavorando in modo asincrono, poiché tutti i rimanenti bit verranno inviati con la stessa durata.

In genere il numero dei bit dei dati varia da 5 a 8. Dopo l'ultimo bit dei dati, può esserci o meno un altro bit detto di parità.

Il bit di parità serve per dare al ricevente un'informazione circa la correttezza del dato ricevuto. La parità può essere pari o dispari. Ultimo bit trasmesso è il bit di stop.

Anche se non canonicamente, abbiamo quindi stabilito implicitamente quali siano le regole del dialogo.

Cerchiamo di visualizzarle in modo più rigoroso, perché il lettore sia in grado di capire più dettagliatamente i discorsi.



Ecco il risultato finale della basetta montata

Velocità di Trasmissione Ricezione

Abbiamo detto che la durata del bit di start è molto importante perché da essa si deduce la velocità di invio del singolo bit. Tale durata in genere viene definita con l'unità BAUD RATE.

Per baud rate si intende il numero di bit per secondo che la porta riesce a gestire. Fino a poco tempo fa erano poche le

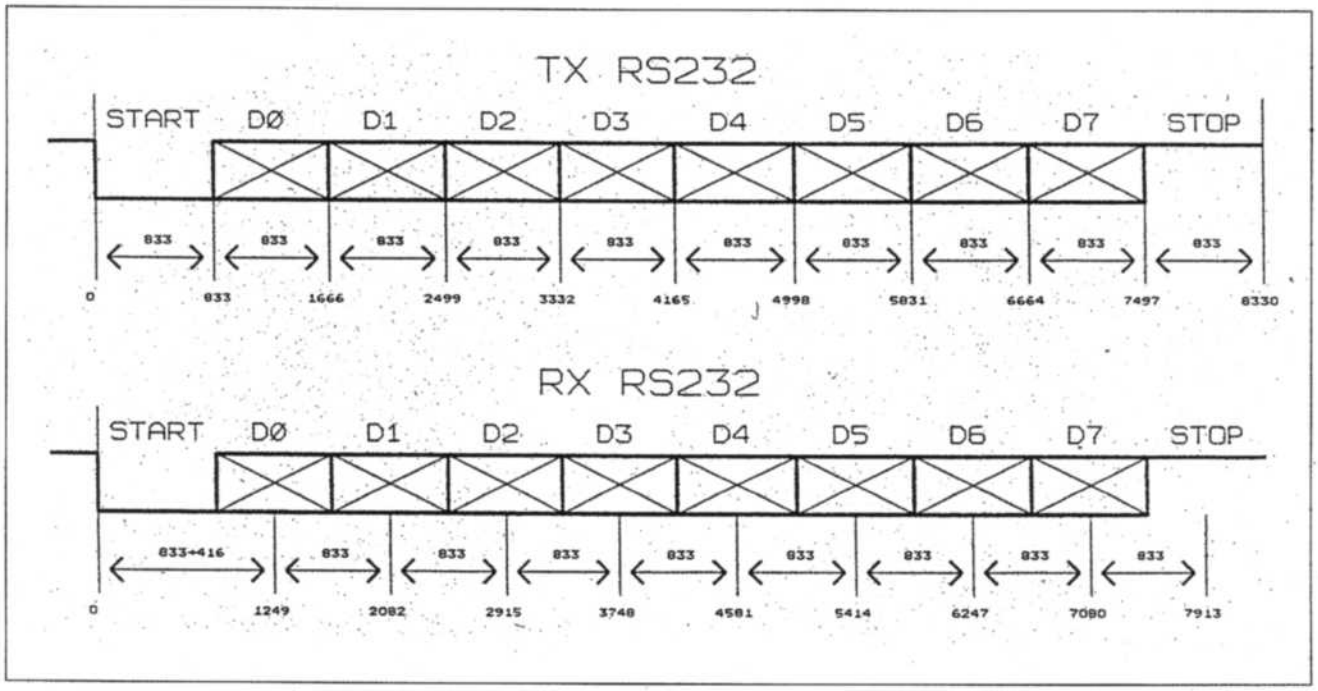


Figura 6. Esempio di trasmissione e ricezione seriale asincrona

velocità raggiungibili, adesso se ne trovano molte di più: 75, 150, 300, 600, 1.200, 2.400, 4.800, 9.600, 14.400, 19.200, 28.800, 38.400 ecc.

Abbiamo accennato che il baud rate è il numero di bit che in un secondo passano potenzialmente dalla porta.

Diciamo potenzialmente perché sulla porta in realtà non scorrono dati in continuazione, quindi non avendo un trasferimento continuo il baud rate medio scenderà di valore.

Allora, per calcolarci la durata di un bit si dovrà scrivere 1/BAUD per ottenere il risultato in secondi.

Vediamolo con un esempio: in figura 6 troviamo lo schema di una trasmissione e di una ricezione seriale asincrona a 1.200 baud 8N1 (poi vedremo che cosa significano questi altri valori). La durata del singolo bit varrà allora: 1/1.200 = 0,000833 secondi, ovvero 833 microsecondi.

Per ottenere la trasmissione, si invia lo start bit con durata di 833 microsecondi, poi si invia il dato D0 per altrettanti 833 microsecondi, poi il dato D1 e così via fino al bit di stop, anch'esso di 833 microsecondi.

Il motivo per cui con la trasmissione seriale asincrona i bit debbono avere tutti medesima durata è l'assenza di un segnale di clock.

In questo modo, partendo dall'arrivo del bit di start, il ricevente può aspettare il tempo adeguato prima di vedere il valore del bit in arrivo. Supponiamo questa volta di essere la sezione ricevente: notiamo il bit di start e, dal fronte di discesa, iniziamo a contare il tempo di arrivo di un singolo bit più quello di un altro mezzo bit.

Così facendo, finita l'attesa, saremo posizionati proprio al centro del primo bit da ricevere (D0) e lo potremo leggere.

Poi dovremmo soltanto attendere la durata di un altro bit intero e leggeremo il bit successivo fino al termine degli otto bit dei dati.

Detto questo, si capisce bene perché sia indispensabile che entrambi i dialoganti conoscano la velocità in baud: se ad esempio il trasmittente

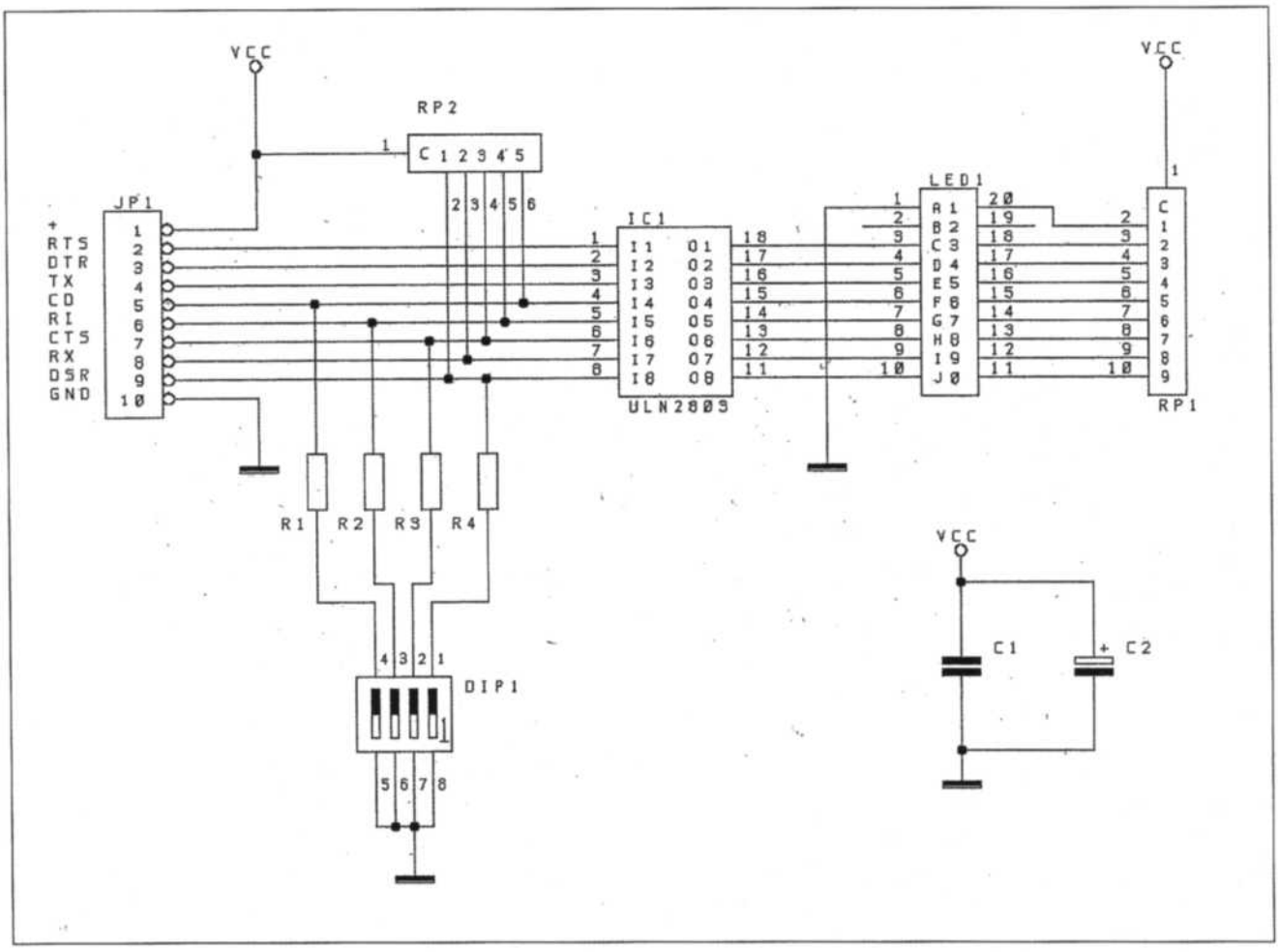


Figura 7. Schema elettrico del circuito di test

invia a 1.200 baud ed il ricevente riceve a 2.400 baud, i vari campionamenti saranno inevitabilmente tutti sfalsati generando errori di lettura.

Allo stesso modo è importante che entrambi i dialoganti posseggano un buon clock interno, per non discostarsi troppo l'uno dall'altro.

Vediamolo con un esempio molto semplice: il trasmittente invia correttamente con buad rate di 1.200.

Il ricevente è selezionato anch'esso per 1.200 baud, ma ha un clock interno che va più lento del 10%.

Ciò significa che il primo bit andrà a campionarlo dopo (833+10%)+(833+10%)/2 cioè dopo 916 + 458 = 1.374 microsecondi. Il primo bit quindi viene letto correttamente (perché compreso tra 833 e 1.666 µs).

l secondo bit verrà campionato a 1.374 + 916 = 2.290 µs ed anche qui tutto va bene (perché compreso tra 1.666 e 2.499).

Il terzo bit verrà campionato dopo 2.290 + 916 = 3.206 microsecondi ed ancora va tutto bene (il terzo bit è compreso tra 2.499 e 3.332). Il quarto bit verrà campionato dopo 3.206 + 916 = 4.122 microsecondi ed anche qui tutto ok (perché compreso tra 3.332 e 4.165 µs).

Il quinto bit invece verrà campionato dopo 4.122 + 916 = 5.038 microsecondi.

Questa volta il dato che leggeremo non sarà però valido, poiché il quinto bit dovrebbe essere compreso tra 4.165 e 4.998 us mentre invece siamo andati oltre.

Il risultato è che al posto del reale quinto bit, leggeremo il sesto e così via shifteranno anche gli ultimi restituendo un dato falsato rispetto a quello inviato.

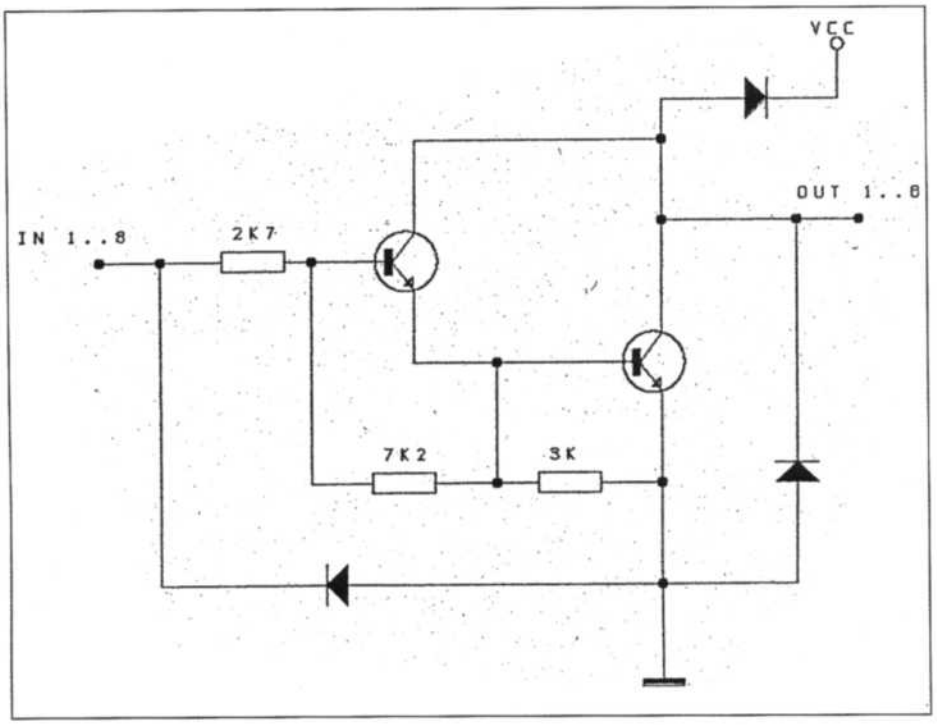
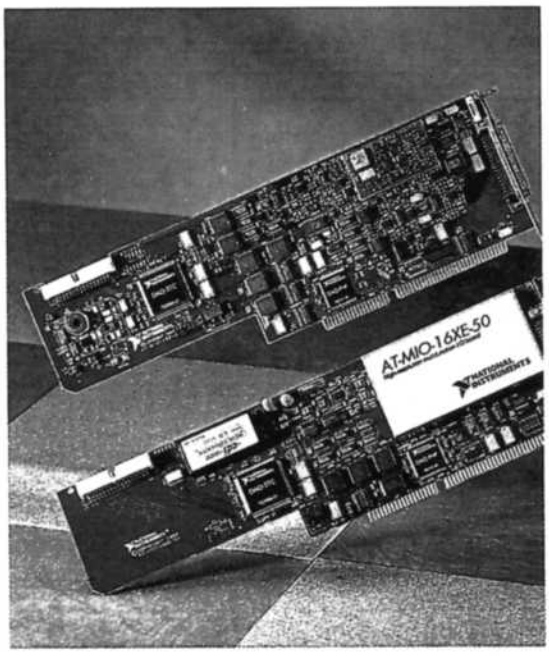


Figura 8. Circuito buffer di IC1

Ci siamo accorti però che ciò avviene con una differenza del 10%, che è relativamente grande come tolleranza per oscillatori quarzati ed in più il danno è avvenuto al quinto bit. Questo è anche uno dei motivi per cui nelle trasmissioni seriali asincrone il numero dei bit che vengono inviati in un frame sono pochi.



BIT di dato

Il numero di bit dei dati è quel numero che indica quanti bit verranno inviati dopo il bit di start inerenti ai dati.

Abbiamo visto che tale numero in genere varia tra 5 e 8.

BIT di parità

Il bit di parità, se presente, viene sempre inviato dopo l'ultimo bit dei dati e prima del bit di stop.

Il bit di parità indica se il numero di "1" presenti nell'intero frame è pari o dispari e per questo possiamo parlare di parità pari o parità dispari.

BIT di stop

Il bit di stop è l'ultimo bit inviato dal trasmittente.

In realtà non è un vero e proprio bit in quanto fisicamente la linea resta in uno stato di inattività.

Serve più che altro a contribuire alla sincronizzazione tra i due interlocutori. Il numero di bit di stop può variare e assumere valori 1, 1,5 o 2.

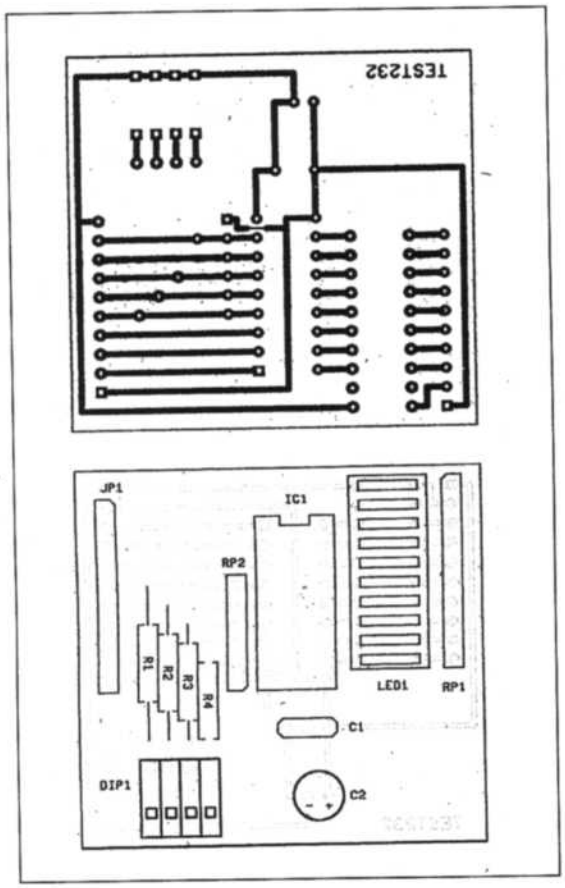


Figure 9 e 10. Circuito stampato e disposizione dei componenti

I parametri

Abbiamo quindi visto quali sono i parametri da conoscere per la comunicazione seriale asincrona.

Quando precedentemente abbiamo detto comunicazione a 1.200 8N1, abbiamo sintetizzato in un modo standard la scrittura dei seguenti parametri: baud-rate = 1.200, numero di bit di dati = 8, parità = nessuna, numero di bit di stop = 1.

Il carattere N della parità può essere sostituito dalle lettere E (EVEN = pari) o O (ODD = dispari).

Le altre linee dedicate

Ma come abbiamo visto lo scorso mese, le linee presenti su una porta seriale sono diverse: impariamo a conoscerle per le loro funzioni. La linea DTR (Data Terminal Ready) è di uscita per la porta ed avvisa il ricevente che il trasmittente è pronto a trasmettere.

La linea TX (Trasmit) è anch'essa di uscita ed è proprio quella dove passano i dati.

La linea RTS (Request To Send) è pure di uscita e comunica che il ricevente è pronto a ricevere.

La linea RX (Receive) è la linea dove attraverso quale ricevuti i dati.

La linea DSR (Data Set Ready) è di input e serve al ricevente per sapere se il trasmittente è pronto o meno a trasmettere.

La linea CTS (Clear To Send) è anch'essa di input e serve al trasmittente per capire se il ricevente è pronto a ricevere.

La linea CD (Carrier Detect) indica se è presente una portante (ad esempio nel caso di un modem).

Infine la linea RI (Ring Indicator) serve per sapere quando, nel caso di un modem collegato alla linea telefonica, si presenta uno squillo telefonico.

Questo avviene per poter poi attivare le procedure di autoanswering (risposta automatica). Passiamo ora ad esaminare un circuito di test.

Il circuito di test

Per poter vedere con l'occhio tutti questi segnali, abbiamo approntato il circuito visibile in figura 7.

Partendo dal connettore JP1, le prime tre linee in alto sono quelle di uscita, le rimanenti sono quelle di ingresso.

Sulle linee di ingresso sono stati inseriti dei dip-switches che consentiranno poi di poter effettuare delle prove simulando lo stato delle linee.

Per quanto riguarda la sezione di visualizzazione, questa è stata implementata con un integrato buffer tipo ULN2803 che pilota una striscia di 10 LED. Ad ogni LED corrisponde un diverso segnale.

Il penultimo LED non è stato collegato mentre l'ultimo serve da spia per la presenza o meno della tensione di alimentazione.

La rete resistiva composta da R1, R2, R3 e R4 e RP1 potrebbe sembrare superflua, in quanto abbiamo visto che l'integrato che ci interfaccia con la porta ha al suo interno delle resistenze di pull-up, ma purtroppo queste non sono di valore sufficiente a far entrare in funzione i buffer di IC1.

In figura 8 infatti vediamo come è composto un buffer dell'ULN2803: lo stadio d'ingresso, che è quello che ci interessa,

ELENCO COMPONENTI

Semiconduttori

IC 1: ULN2803

Resistori

R1 ÷ R4 : 100 Ω

RP 1: Rete resistiva 220 Ω 9 + 1

RP 2 : Rete resistiva 10 kΩ 5 +1

Condensatori

C1: 100 nF

C2: 47 µF

Varie

LED 1: Barra di led x 10

DIP 1 : Dip-switch x 4

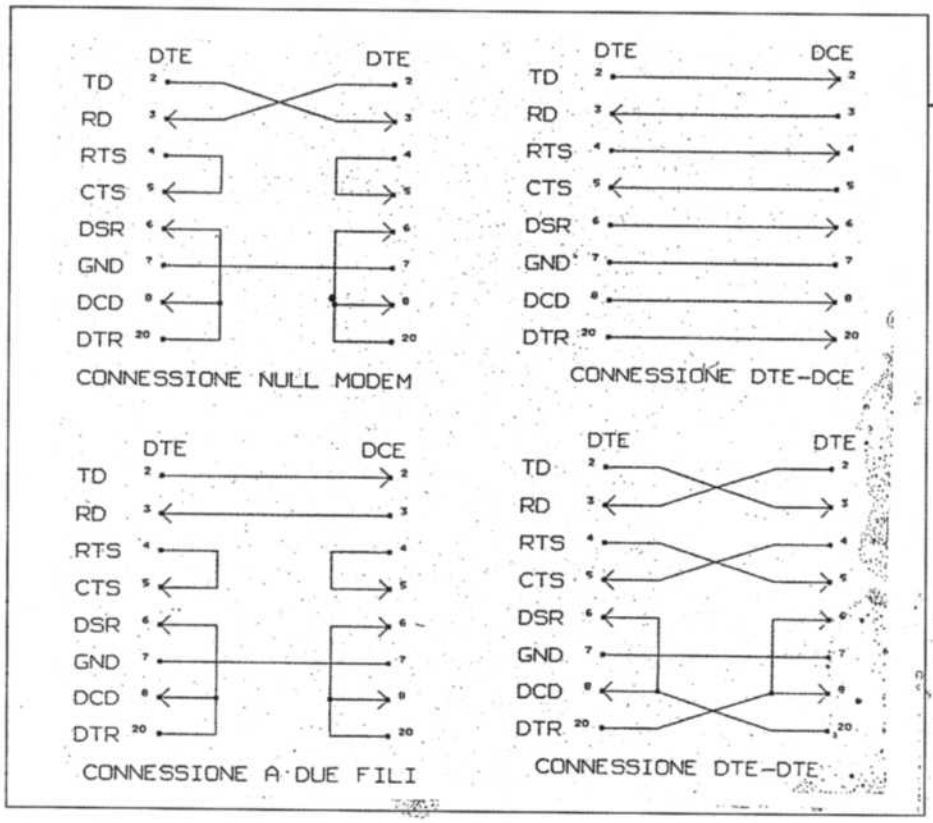


Figura 11. Collegamenti seriali standard

della barra di LED, pena il non funzionamento dell'intera interfaccia.

L'alimentazione di 5 volt viene prelevata direttamente dal circuito dell'interfaccia presentata precedentemente.

Per quanto riguarda il collaudo, ognuno potrà farlo a modo proprio, con il software di cui dispone.

Ad esempio, si potrà verificare che senza alcuni segnali di convalida la seriale si rifiuta di inviare o di ricevere dei dati.

Per concludere, in figura 11 troviamo i collegamenti tipici per cavetti seriali standard mentre in tabella 2 i segnali relativi sia alla presa DB9, sia alla presa DB25. ■

ha una resistenza in serie da 2,7 kΩ ed altri 10 kΩ verso massa.

Per far accendere i diodi led correttamente, si è reso quindi necessario inserire la rete resistiva RP2.

Montaggio e collaudo

Per questa realizzazione è sufficiente un circuito stampato a faccia singola e la cui traccia è visibile in figura 9.

In figura 10 invece troviamo il piano di cablaggio di tutti i componenti da inserire sulla schedina.

Per quanto riguarda JP1, consigliamo di impiegare un connettore maschio della lunghezza minima di 1cm, per poterlo adattare al connettore femmina relativo al circuito del mese scorso.

Si faccia molta attenzione al verso delle reti resistive e

Tabella 2 - Connessioni su porte DB9 e DB25

DB25	DB9	Segnale	Funzione	DTE	DCE
1		CG	Chassis ground	–	–
2	3	TD	Trasmissione dati	out	in
3	2	RD	Ricezione dati	in	out
4	7	RTS	Request To Send	out	in
5	8	CTS	Clear To Send	in	out
6	6	DSR	Data Set Ready	in	out
7	5	SG	Signal Ground		
8	1	DCD	Data Carrier Detect	in	out
9	–	–	Positive test voltage	–	–
10	–	–	Negative test voltage	–	–
11	–	–	Non utilizzato	–	–
12	–	SDCD	Secondary DCD	in	out
13	–	SCTS	Secondary CTS	in	out
14	–	STD	Secondary TD	out	in
15	–	TC	Transmit Check	in	out
16	–	SRD	Secondary RD	in	out
17	–	RC	Receive Clock	in	out
18	–	–	Non utilizzato	–	–
19	–	SRTS	Secondary RTS	out	in
20	4	DTR	Data Terminal Ready	out	in
21	–	SQ	Signal Quality Detect	in	out
22	9	RI	Ring Indicator	in	out
23	–	SEL	Speed selector DTE	in	out
24	–	TCK	Speed selector DCE	out	in
25	–	BSY	Data line busy	in	out