

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

Interfaccia tra Centralino Telefonico e PC parte prima : gli aspetti dell'Hardware

Realizzato da :

ROLFI FABRIZIO

STANGA FABIO

FALETTI MASSIMO

classe 5BI TIEE

corso per Tecnici delle Industrie Elettriche ed Elettroniche

anno scolastico 1996-97

Indice :

OBIETTIVO DELLA PROVA	3
<i>Schema a blocchi.</i>	4
<i>Descrizione del cavo di collegamento</i>	4
<i>Schema del cavo di interfaccia</i>	5
TECNICHE DI INTERFACCIAMENTO	5
<i>Schema di interfacciamento parallelo</i>	6
<i>Temporizzazione dei segnali nel caso di interfaccia parallela</i>	6
<i>Interfaccia parallela PC : lato stampante</i>	7
<i>Interfaccia parallela PC : lato PC (DB25F)</i>	8
INTERFACCIAMENTO SERIALE	9
L'INTERFACCIA RS-232C STANDARD E LE SUE APPLICAZIONI	11
<i>Il pinout dei connettori RS-232</i>	12
<i>Livelli di Tensione dell'RS-232</i>	14
<i>Connettori RS-232C standard a 9 pin</i>	14
SISTEMI DI COMUNICAZIONE SERIALE SINCRONA E ASINCRONA	15
SINCRONIZZAZIONE DI UN RICEVITORE ASINCRONO	17
CIRCUITI INTEGRATI UTILIZZATI NELLE INTERFACCE RS-232	18
DESCRIZIONE GENERALE DELLA UART NATIONAL INS8250	20
<i>Descrizione dei pin dell'integrato INS8250 (lato EIA driver)</i>	21
<i>Segnali di ingresso</i>	21
<i>Segnali di Uscita</i>	22
<i>LCR (Line Control Register/Registro delle linee di controllo)</i>	22
<i>LSR (Line Status Register/Registro di stato delle linee)</i>	24
<i>MSR (Line Modem Register)</i>	25
<i>MCR (Modem Control Register)</i>	26
INTERFACCIA "CURRENT LOOP" (LOOP DI CORRENTE)	27
STAMPANTE EPSON LX- 800	28
<i>Interruttore di attivazione</i>	28
<i>Dip Switch</i>	28
<i>Velocita' di trasmissione</i>	29
<i>Connettore e Segnali</i>	30
<i>Trasmissione Dati</i>	30
<i>Il Buffer</i>	30
<i>Buffer disabilitato</i>	31
<i>Buffer abilitato</i>	31
<i>Controllo tramite flag</i>	31
<i>Controllo tramite protocollo XON/XOFF</i>	31
<i>Test</i>	32
<i>Loopback</i>	32
<i>Line monitor</i>	32
<i>Programmazione Stampante Epson Lx-800</i>	33
PROGRAMMAZIONE IN TURBO PASCAL 6.0	34
<i>Tipi di Dati</i>	35
<i>Libreria di Gestione della UART 8250 in Turbo Pascal 6.0</i>	36

Premessa e obiettivo della Prova

Nell' istituto è presente un centralino telefonico corredato di interfaccia seriale connessa ad una stampante, in grado di riportare su carta un resoconto dettagliato di tutte le chiamate effettuate. Il sistema riporta su di un tabulato il numero del telefono interno da dove è partita la telefonata, il numero composto dall'utente, la linea del centralino impegnata, quanti scatti ha effettuato, l'ora in cui ha chiamato e infine la durata della telefonata.

Il dispositivo da noi realizzato intercetta i segnali seriali che il centralino invia alla stampante; li riaorganizza in un file e li memorizza su disco.

Dal punto di vista hardware è stato necessario realizzare un apposito cavo seriale in grado di derivare i segnali necessari dalla connessione centralino/stampante.

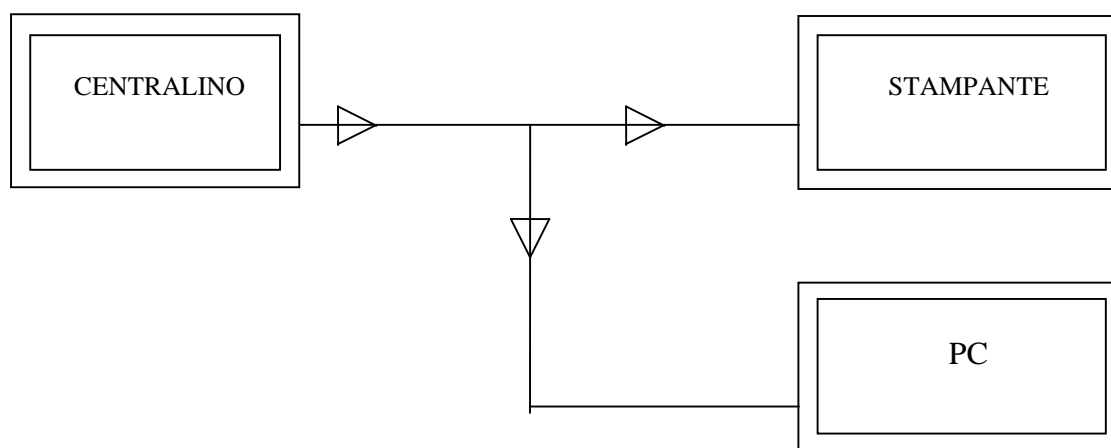
Questo tipo di esperimento effettuato per risolvere un problema interno all'istituto, ha offerto lo spunto per studiare e approfondire le problematiche relative agli interfacciamenti in genere che si possono effettuare fra un PC ed un sistema esterno attraverso l'utilizzo di dispositivi cosiddetti seriali o paralleli.

Gli studenti, coordinati dal docente hanno raccolto una serie di informazioni riguardanti l'argomento, hanno tradotto dall'Inglese buona parte dei data-sheet del chip 8250 UART presente nei PC di tipo XT/AT ed hanno predisposto una piccola libreria in Turbo Pascal per la gestione dell'UART.

Un secondo gruppo il prossimo anno scolastico predisporrà il software di generazione degli archivi ed il software per l'analisi statistica dei dati in essi contenuti.

Giugno 97

prof. Cleto Azzani

Schema a blocchi.

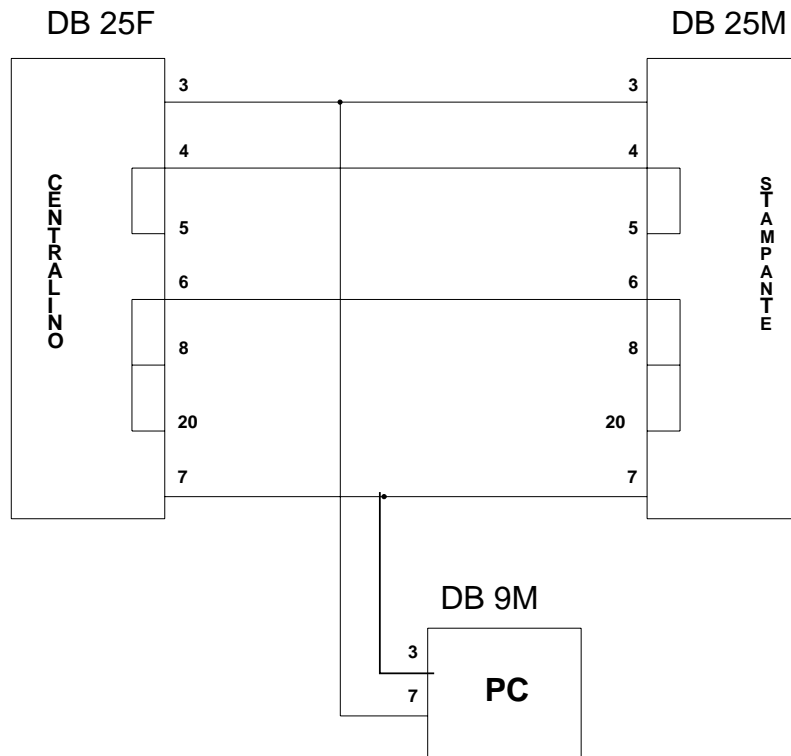
Un programma scritto in TURBO PASCAL ver.6.0 ha consentito al personal computer interposto, tramite lo speciale cavo seriale da noi realizzato, di acquisire e gestire tutte le informazioni che costituiscono l'oggetto della nostra prova.

Descrizione del cavo di collegamento

Il cavo da noi realizzato prevede il collegamento tra la stampante EPSON LX-800 e il centralino telefonico della TELECOM, da questa connessione vi è una derivazione con un cavo al nostro PC. La costruzione del cavo è avvenuta rispettando le lunghezze standard non superiore a 15 metri la quale porterebbe alla nascita di capacità parassite molto elevate che creerebbero errori di trasmissione. Le interfacce utilizzate sono seriali modello RS-232, il DB25F (modello a 25 poli femmina) per il centralino telefonico, il DB25M (modello a 25 poli maschio) per la stampante seriale EPSON e infine il DB9M (modello a 9 piedini maschio) per il PC 80286 COMPAQ. Per l'interfacciamento tra il centralino e la stampante i pin da utilizzare sono stati copiati tali e quali dai collegamenti presenti sull'interfaccia DB25F del centralino e sono stati riportati sull'interfaccia DB25M della stampante: il pin 3 (RXD), il ponte tra il pin 4 (RTS) e il pin 5 (CTS), il ponte tra il pin 6 (DSR), il pin 8 (DCD) e il pin 20 (DTR), infine il pin 7 di massa (GND).

Dalla derivazione effettuata per il PC sono stati utilizzati solamente il pin 3 e il pin 7 sufficienti a soddisfare le nostre esigenze. I pin dell'interfaccia seriale e la loro funzione verranno approfonditi dettagliatamente nel corso dell'elaborato. Il cavo da noi realizzato è mostrato in figura.

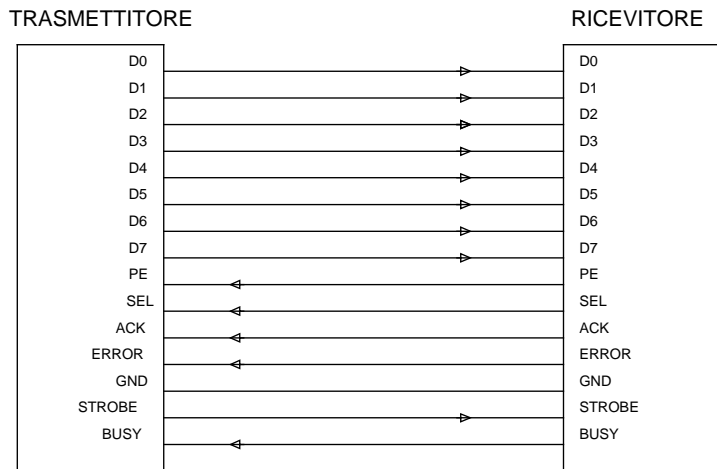
Schema del cavo di interfaccia



Tecniche di Interfacciamento

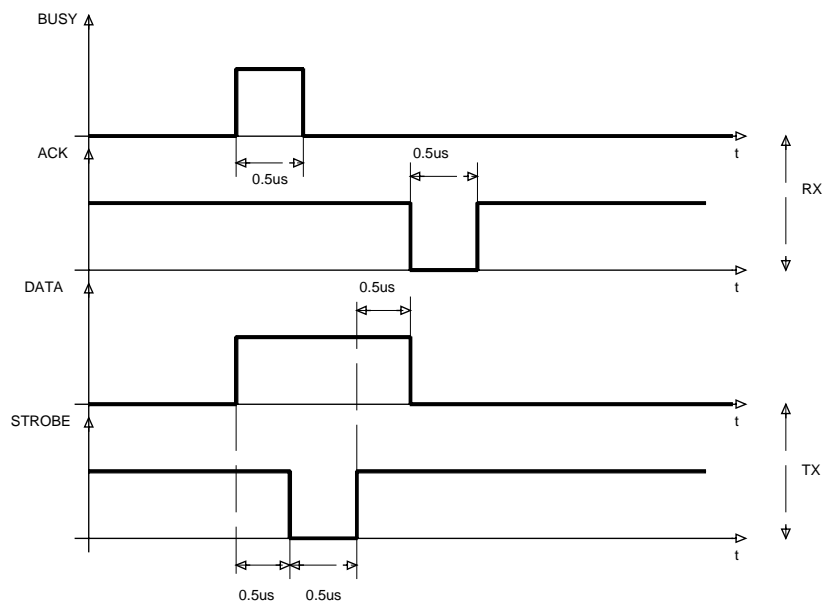
I sistemi di interfacciamento, vengono utilizzati nei PC per connettere il sistema di elaborazione dati con i dispositivi periferici esterni (tastiera, mouse, video, stampante, plotter, unità HD, FD, CD-ROM, modem etc). Alcuni sistemi di interfacciamento sono essenziali per il funzionamento del sistema stesso (interfaccia video, interfaccia tastiera), altri sono opzionali. Fra le modalità di trasferimento dati distinguiamo la “modalità parallelo” attuata ad esempio nel caso dell’interfaccia di una stampante e interfacciamento seriale (verso un modem o un plotter o una stampante seriale). Nel caso di interfaccia parallela, il dato a 8 bit che deve essere trasferito fra sistema di elaborazione dati e periferica esterna transita su otto conduttori distinti che costituiscono un “bus” (veicolo di trasmissione dei dati). Il dato in questo caso viene trasmesso in un unico ciclo contemporaneamente sugli otto conduttori distinti che costituiscono il “bus”.

Schema di interfacciamento parallelo



Temporizzazione dei segnali nel caso di interfaccia parallela

Nella figura a è mostrato il comportamento dei segnali nel tempo. Una volta partito il segnale con un brevissimo ritardo approssimativamente di 0.5 usec, parte il segnale di strobe indicante che la stampante è attiva, il ritardo permette ai dati di arrivare tutti contemporaneamente. All'interno della stampante c'è un buffer (che corrisponde ad una riga) che richiede tempo per riempirsi e viene svuotato quando è pieno. L'operazione consecutiva al comando di strobe è la segnalazione di ACKNOWLEDGE (ACK), un segnale che normalmente è attivo a 1 (nel momento in cui invio il dato e lo strobe) e va a livello 0 quando la stampante ha finito di stampare il dato (buffer vuoto) quindi è pronta a ricevere altri dati, l'impulso dura circa 0.5 usec. Un'altra linea di controllo è il BUSY, un segnale che normalmente è a livello 0 (prima del dato) e poi quando arriva il dato passa a livello attivo 1, questo impulso indica che la stampante non è in grado di ricevere dati.



Nella seguente tabella sono riportati alcuni pin dell'interfaccia parallela con le relative funzioni.

Interfaccia parallela PC : lato stampante

Denominazione del pin	I/O	Note
<i>D0,D1,D2,D3.....D7</i>	<i>input</i>	<i>Bit di ingresso</i>
<i>Strobe</i> (Dato Valido)	<i>input</i>	<i>Il livello di questo segnale normalmente è alto, si abbassa quando il PC invia dei dati. Siccome i bit che viaggiano sui cavi conduttori non giungono a destinazione tutti contemporaneamente, il segnale di "strobe" avverte il ricevitore che i dati sulle 8 linee sono validi. Il segnale di strobe viene generato in ritardo rispetto alla trasmissione dei dati.</i>
<i>Acknowledge</i> (Dato Ricevuto)	<i>output</i>	<i>Un livello basso di acknowledge, indica che la stampante è pronta a ricevere altri dati, l'impulso è di circa 0,5 usec.</i>
<i>PE</i> (Mancanza Carta)	<i>output</i>	<i>Un livello alto del PE indica che la carta nella stampante è terminata.</i>
<i>Error</i>	<i>output</i>	<i>Indica che la stampante non è in linea o si trova in stato di errore.</i>
<i>Gnd</i>	<i>-----</i>	<i>Ritorno di massa (Signal ground).</i>
<i>Busy</i> (Stampante Occupata)	<i>output</i>	<i>Un livello alto di Busy indica che la stampante non è in grado di ricevere dati.</i>

Connettore vaschetta 25 poli maschio sul cavo stampante connesso al PC
Connettore Centronics 36 poli verso la stampante

Interfaccia parallela PC : lato PC (DB25F)

Denominazione del pin	I/O	Note
<i>D0,D1,D2,D3.....D7</i>	<i>output</i>	<i>Bit di ingresso</i>
<i>Strobe</i> (Dato Valido)	<i>output</i>	<i>Il livello di questo segnale normalmente è alto, si abbassa quando invia dei dati. Siccome i bit che viaggiano sui cavi conduttori non giungono a destinazione tutti contemporaneamente, il segnale di "strobe" avverte il ricevitore che i dati sulle 8 linee sono validi. Il segnale di strobe viene generato in ritardo rispetto alla trasmissione dei dati.</i>
<i>Acknowledge</i> (Dato Ricevuto)	<i>input</i>	<i>Un livello basso di acknowledge, indica che la stampante è pronta a ricevere altri dati, l'impulso è di circa 0,5 usec.</i>
<i>PE</i> (Mancanza Carta)	<i>input</i>	<i>Un livello alto del PE indica che la carta nella stampante è terminata.</i>
<i>Error</i>	<i>input</i>	<i>Indica che la stampante non è in linea o si trova in stato di errore.</i>
<i>Gnd</i>	<i>-----</i>	<i>Ritorno di massa (Signal ground).</i>
<i>Busy</i> (Stampante Occupata)	<i>input</i>	<i>Un livello alto di Busy indica che la stampante non è in grado di ricevere dati.</i>

Connettore vaschetta 25 poli femmina sul PC

Connettore vaschetta 25 poli maschio sul cavo stampante

Interfacciamento Seriale

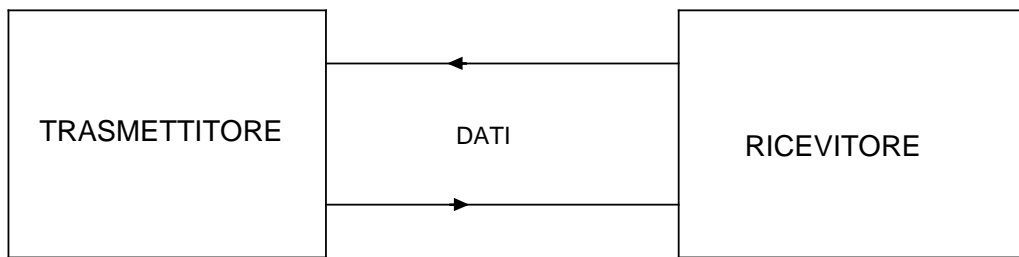


figura 1

In un interfacciamento seriale i bit vengono trasferiti uno per volta, a differenza dell'interfacciamento parallelo in cui viene trasferito un byte per volta. Il modello di interfaccia seriale mostrato in figura 2 è costituito da due registri digitali chiamati shift register (registri di scorrimento). Il registro di scorrimento PISO riceve i dati provenienti da una sorgente digitale parallela e li trasforma in una stringa di bit serializzati prelevabile sull'uscita del registro. La frequenza di clock determina la velocità di trasmissione del registro a scorrimento, ossia il numero di bit emessi dall'uscita nell'unità di tempo. E' possibile perciò attraverso gli ingressi del PISO caricare entro il registro un determinato messaggio proveniente da una sorgente dati parallela; il segnale load abilita l'operazione di caricamento. Il segnale di clock provoca lo scorrimento dei dati che fuoriescono in modo seriale dall'uscita dello shift register. Lo shift register PISO riceve i dati dal registro a scorrimento PIPO, dopo 8 periodi di clock i dati disponibili in uscita al PIPO possono essere caricati all'interno del PISO, ciò avviene tramite il comando di load. Il registro PIPO è abilitato dal comando di load, che carica i dati all'ingresso del registro e li rende disponibili in uscita. In cascata allo shift register PISO, è collegato un registro a scorrimento di tipo SIPO, il quale riceve i segnali provenienti da una sorgente digitale seriale e li trasforma in una stringa di bit disponibili in parallelo sulle sue uscite. L'informazione è disponibile sulle uscite parallele solo dopo che lo shift register ha ricevuto 8 periodi di clock. Il dato si conserva integro solo per un periodo di clock. Il dispositivo ricevente deve perciò provvedere con la massima tempestività a leggere il dato dal SIPO. Visto che il messaggio si conserva solo per un periodo, se la lettura avviene in ritardo il dato si deteriora, perciò è stato inserito un registro di tipo PIPO che riceve in entrata segnali paralleli e in uscita restituisce altrettanti segnali paralleli. Il PIPO è un registro che serve per tenere memorizzato il dato per 8 clock (ciò è permesso dal contatore per 8), dopo di che il dato si è deteriorato e viene sovrascritto da altri dati provenienti dal SIPO che vengono caricati dal comando di load. La porta NOT è stata inserita per fare in modo che la memorizzazione nello SR di ricezione avvenga in ritardo di mezzo periodo rispetto all'operazione

di shift nel registro PISO di trasmissione. Ciò permette al dato di assestarsi e al ricevitore (SIPO) di non essere influenzato dai transitori sempre presenti sulla linea dei dati.

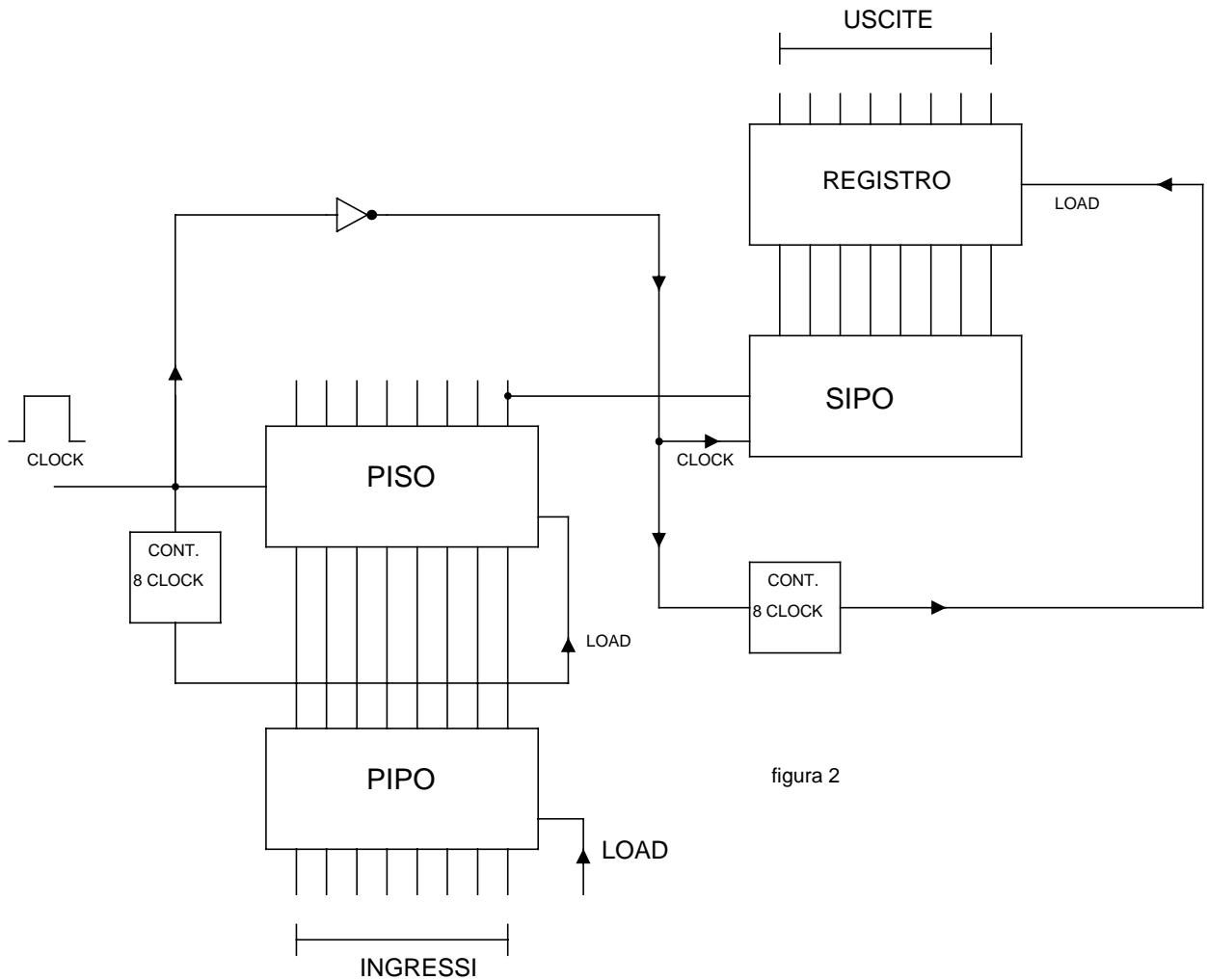
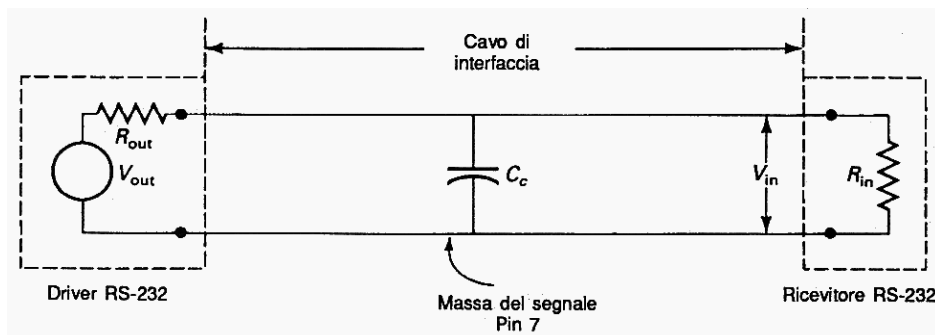


figura 2

L'interfaccia RS-232C standard e le sue applicazioni

Lo standard RS-232 (RS significa standard raccomandato) è stato introdotto negli anni sessanta per facilitare la compatibilità e l'interscambiabilità tra PC e dispositivi periferici collegati mediante porte seriali. Esaminiamo ora una applicazione schematica nel campo delle comunicazioni. La figura a mostra un modello semplificato di un'interfaccia per telecomunicazioni, composto da un trasmettitore RS-232 e un ricevitore RS-232. Il trasmettitore viene denominato driver (pilota) e il ricevitore viene anche chiamato terminatore. Passiamo ora all'analisi di ciascun elemento.



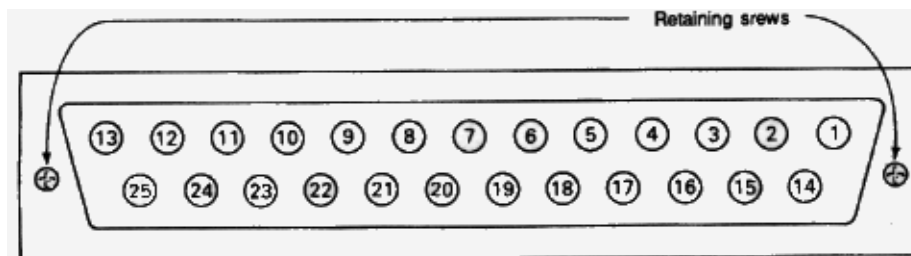
- **V_{out} (tensione in uscita del circuito aperto del driver):** questo valore è il MARK o lo SPACE emessi dal trasmettitore in una condizione ideale di non carico.
- **R_{out} (resistenza in uscita del driver):** in un circuito ideale questa resistenza vale 0Ω , tutti i driver possiedono una resistenza misurabile in uscita; con l'aumento della corrente in un driver, le tensioni di uscita diminuiscono in relazione al valore della R_{out} secondo la legge di Ohm.

$$V_{IN} = V_{OUT} - I_{OUT} \cdot R_{OUT}$$

- **C_c (capacità del cavo di trasmissione):** la capacità della linea provoca disturbi sulla forma d'onda del segnale, questa capacità rappresenta il fattore che porta alla limitata lunghezza del cavo dell'interfaccia RS-232 (max 15 m).
- **R_{in} (resistenza di entrata del ricevitore):** idealmente questa resistenza assume valore infinito, la maggior parte della tensione viene assorbita da questa resistenza e compare come V_{in} per il ricevitore. La R_{in} deve essere in grado di assorbire i massimi valori di tensione dell'RS-232 (+25 e -25) senza che il ricevitore subisca danni.

Il pinout dei connettori RS-232

I pinout del connettore RS-232 consistono nella disposizione dei pin, la figura a illustra il collettore RS-232 a 25 pin. Sempre facendo riferimento a un collegamento modem computer, descriviamo i vari pin con le relative funzioni.



I pin descritti possono essere suddivisi in tre gruppi: MASSE, DATI e HANDSHAKING.

MASSE

- **PIN 1** (massa della struttura): è collegata alla struttura del terminale o a terra, spesso viene lasciata sconnessa. Questa massa serve a isolare i terminali da eventuali tensioni pericolose, a eliminare i disturbi che possono essere creati dalla struttura dell'apparecchio qualora si trovi a terra.
- **PIN 7** (massa del segnale): tutti i segnali dell'interfaccia RS-232 sono riferiti alla massa del segnale, esso deve essere continuamente presente per garantire il corretto funzionamento del circuito.

DATI

- **PIN 2** (trasmissione dati TD): i dati vengono trasferiti dal terminale o computer al modem attraverso questo pin.
- **PIN 3** (ricezione dati RD): i dati vengono trasmessi dal modem al terminale attraverso questo pin.

HANDSHAKING

Questo termine indica la cooperazione tra i dispositivi che stanno comunicando o scambiando dati, questi dispositivi devono indicare lo stato del proprio trasmettitore e ricevitore.

- **PIN 4** (richiesta di trasmissione RTS): viene portato a livello attivo stando a indicare che il terminale o computer è pronto alla trasmissione dei dati. La risposta del modem è quella di prepararsi alla ricezione.
- **PIN 5** (clear to send CTS): questo pin serve al modem per indicare al computer che la trasmissione dei dati può cominciare, quindi il modem porta il CTS a livello attivo. Il pin 4 e 5

funzionano contemporaneamente, il PC invia un RTS , dopo un breve ritardo il modem risponde con un livello attivo di CTS.

- **PIN 20** (terminale dati pronto DTR): il DTR viene attivato dal terminale o computer al momento dell'accensione. Serve per indicare al modem di essere collegato a un pc.
- **PIN 6** (data set pronto DSR): è il corrispondente del DTR (pin 20) quando il DSR è attivato, il terminale è informato di essere collegato a un modem.
- **PIN 8** (data carrier detect DCD): questo pin denominato riconoscimento portante, viene utilizzato per comunicare al terminale di essere collegato al computer remoto e di poter quindi iniziare la trasmissione dei dati. Il DCD informa il computer della avvenuta esecuzione di un collegamento remoto. I pin non elencati non sono di utilizza comune.

Tabella riassuntiva riferita al PC.

Segnale	PIN	I / O	Descrizione
Massa di sistema	1	-----	Massa a terra dello chassis
TXD (transmitted data)	2	Out	Trasmissione dati
RXD (recived dati)	3	In	Ricezione dati
RTS	4	Out	Richiesta di trasmissione
CTS (clear to send)	5	In	il modem indica che la trasmissione può cominciare
DSR (data set ready)	6	In	Deve essere a livello basso perchè la stampante possa ricevere dati
Massa del segnale	7	-----	Massa del segnale
DCD (data carrier in detect)	8	In	Equivalente a DSR pin 6
RTS (reserv chanel)	11	Out	Deve essere a livello basso perchè la stampante possa ricevere dati
TTY - TXD	17	Out	Una bassa impedenza tra i punti 17 e 24 o uno X on sugli stessi pin
DTR (data terminal out ready)	20	Out	Deve essere a livello basso perchè la stampante possa ricevere dati
TTY - RXD return	23	-----	-----
TTY - TXD return	24	-----	Indica che la stampante è pronta a ricevere dati
TTY -RXD	25	In	Ricezione dati in current loop

Livelli di Tensione dell'RS-232

La tabella che segue indica i livelli di tensione dell'RS-232 e le relative interpretazioni standard.

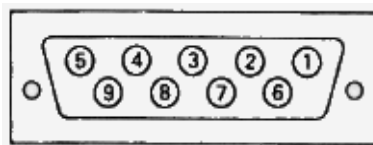
Tensione	Logico	Controllo	Terminologia telescriventi
Da +3V a +25V	0	On	Space
Da -3V a -25V	1	Off	Mark

La tensione positiva nel circuito (da +3V a +25V) rappresenta un valore logico zero, mentre quella negativa (da -3V a -25V) un valore 1. Il valore di tensione compreso tra +3V e -3V è un livello indeterminato. La terminologia standard per indicare i livelli di tensione sono stati adottati dalla terminologia delle telescriventi, per tanto un livello logico zero è detto SPACE, mentre un livello logico 1 è detto MARK. Un RS-232 che non sta trasmettendo nessun segnale mantiene la propria linea di trasmissione a livello logico 1 (mark), quando il segnale viene trasmesso, la linea di trasmissione passa a livello logico zero (space), la linea così subisce una variazione di tensione, da un livello inattivo (da -3V a -25V) a un livello attivo (da +3V a +25V).

Nella maggioranza dei casi un dispositivo a interfaccia RS-232, lavora con alimentazioni da +12V a -12V. Un livello di mark viene espresso da valori compresi da -9V a -12V e uno space in valori compresi tra +9V a +12V.

Connettori RS-232C standard a 9 pin

Poiché per la maggior parte delle applicazioni vengono utilizzati solo 9 dei 25 pin dell'interfaccia RS-232, sono stati sviluppati connettori RS-232 a 9 pin. Questa interfaccia permette di risparmiare spazio nel pannello posteriore del computer, questo tipo di connettore è standard. Il pin 9 che corrisponde al pin numero 22 del connettore a 25 pin, è chiamato ring indicator RI, esso passa a livello di space quando il modem riceve un segnale di chiamata, la funzionalità degli altri pin è identica a quella precedentemente descritta. La figura a mostra un connettore a 9 pin.



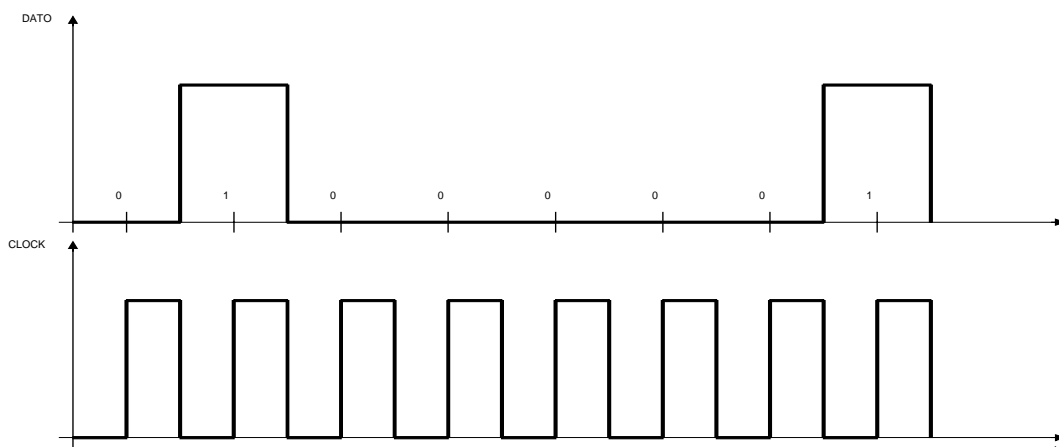
Vista frontale

Tabella dei Pin.

Pin	NOME STANDARD	CORRISPONDENTE DB25
1	DCD	8
2	RD	3
3	TD	2
4	DTR	20
5	SG	7
6	DSR	6
7	RTS	4
8	CTS	5
9	RI	22

Sistemi di comunicazione seriale sincrona e asincrona

Esistono due tipi di comunicazione seriale: *sincrona* ed *asincrona*. Nella comunicazione *sincrona*, oltre ai dati viene trasmesso anche il segnale di clock che serve alla loro codificazione. Questo tipo di comunicazione, viene usata dove sono richieste elevate velocità di trasferimento dati, poiché l'efficienza della trasmissione è molto elevata. Ma da una elevata trasmissione di dati nasce la difficoltà di gestire la perfetta sincronizzazione sul ricevitore, per questo non è facilmente applicabile. Nella figura a è riportato un esempio di comunicazione sincrona di un carattere di lunghezza 8 bit (01000001), da notare è l'andamento del segnale di clock il quale ha un fronte di salita al centro di ogni bit, ciò permette di avere una lettura precisa.



Di più facile applicazione è la comunicazione *asincrona* (da noi trattata specificatamente), la quale elimina la necessità di trasmettere il segnale di clock poiché i caratteri vengono trasmessi uno alla volta e la sincronizzazione avviene su ogni carattere. Nella figura b è riportato un tipico andamento di trasmissione asincrona.



La durata temporale di ogni singolo bit è determinata dalla velocità di trasmissione del dato (baud rate) che è espressa in bit al secondo, l'ampiezza di un bit è espressa dalla formula $1/\text{baud rate}$. Quando la linea non trasmette nessun dato, viene lasciata in uno stato di 1 logico (MARK). Quando invece si vuole trasmettere un dato, lo si fa precedere da un bit di START, rappresentato da un livello logico 0 (SPACE), dopodichè viene trasmesso il dato di lunghezza prestabilita minimo 5 bit, massimo 8 bit, in coda stanno l'eventuale bit di parità e il bit di stop, vista la semplicità della trasmissione asincrona è molto diffusa soprattutto nelle interfacce.

- **BIT DI START:** questo bit precede la trasmissione asincrona ed è costituito da un livello logico 0 di durata pari a quella di un bit.
- **BIT DI PARITA':** durante la trasmissione seriale si possono venire a generare errori sul canale di trasmissione, la lettura inesatta anche di un solo bit può alterare l'intero file di dati. Per ridurre gli errori, si inserisce il bit di parità tra il bit di dati e il bit di stop. Questo bit viene generato in base al numero di 0 e 1 logici contenuti nei dati. La parità può essere pari (even) o dispari (odd). la parità pari setta il bit di parità ad un livello logico che garantisce che il numero totale di "1" livelli logici trasmessi sia pari, mentre a sua volta la parità dispari setta il bit di parità a un livello logico che garantisce che il numero totale di "1" logici trasmessi sia un dispari. Prima della trasmissione dei dati, come per il baud rate, il trasmettitore e il ricevitore, devono concordare il tipo di parità da utilizzare.

Esempio di bit di parità pari (even)

Dato/carattere	Valore esadecimale	Valore binario	Bit di parità
10	\$A	01000001	0
12	\$C	01000011	1

Esempio di bit di parità dispari (odd)

Dato/carattere	Valore esadecimale	Valore binario	Bit di parità
11	\$B	1011	0
15	\$F	1111	1

- **BIT DI STOP:** indica che tutti i dati sono stati trasmessi e la trasmissione del carattere è completata, può assumere valori di 1, 1/2 e 2, ciò dipende dagli standard di costruzione.

Sincronizzazione di un ricevitore asincrono

Un ricevitore asincrono è strutturalmente costituito da uno shift register SIPO che deve opportunamente “leggere” la stringa di bit che giunge dal trasmettitore. La frequenza di clock dello shift register di trasmissione deve necessariamente essere identica a quella dello shift register di ricezione. Ma nel ricevitore asincrono non esiste segnale di clock che assicura la sincronizzazione del ricevitore al trasmettitore. Come si provvede alla ricezione di un carattere ?

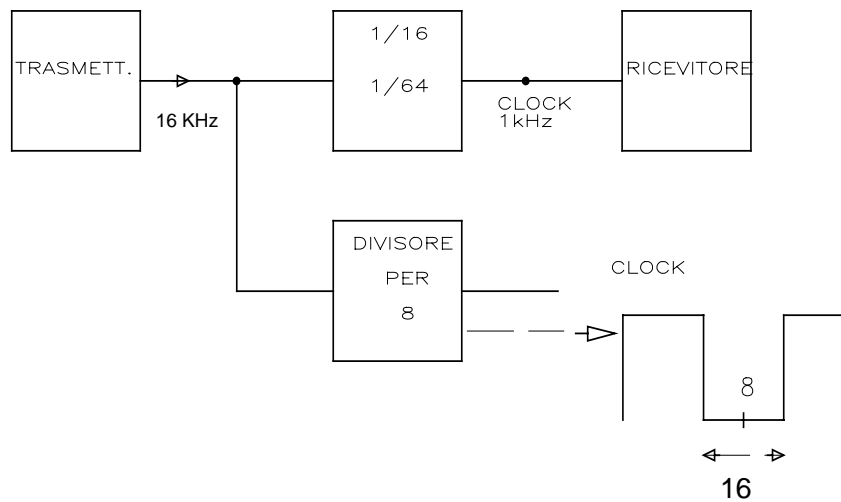
1) La frequenza di clock del SIPO (ricevitore) viene ottenuta da un oscillatore locale al quarzo di frequenza pari a 16 o 64 volte il baud rate. Ciò significa che la ricezione di un bit impegna 16 o 64 clock dell'oscillatore locale.

2) Quando il ricevitore “avverte” il passaggio da 1 a 0 sul suo ingresso seriale inizia il ciclo di ricezione.

- Un apposito divisore per 8 conta 8 periodi di clock del generatore locale ciò garantisce di posizionare la letteratura del SIPO di ricezione esattamente sul tratto centrale del bit trasmesso.
- Vengono letti tutti i bit appartenenti al messaggio trasmesso che sono posizionati a distanza di 16 o 64 bit da quello precedente.
- La lettura si conclude con l'acquisizione dei bit di parità, dei bit di stop.

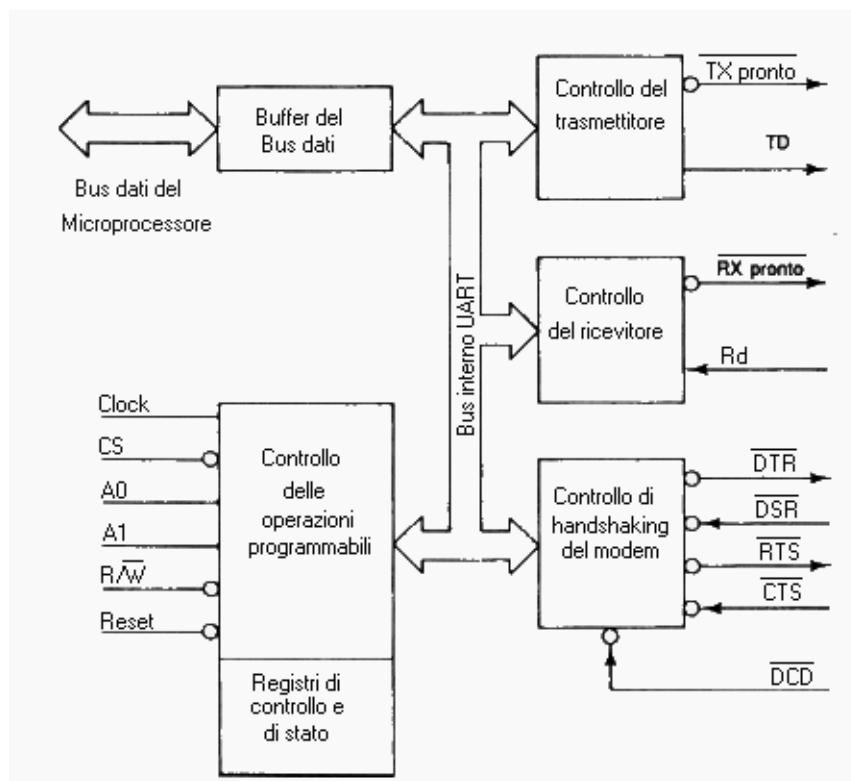
Il ciclo si ripete per la ricezione dei caratteri successivi.

In figura è mostrato in forma semplificata lo schema a blocchi di un circuito ricevitore asincrono.



Circuiti integrati utilizzati nelle Interfacce RS-232

I maggiori risultati sulla semplificazione di terminali, stampanti, e strutture di interfaccia per computer, sono stati raggiunti con l'introduzione del UART (trasmettitore e ricevitore asincrono universale). La maggior parte delle caratteristiche programmabili degli UART includono la velocità di trasmissione, numero di bit di dati, bit di start, bit di stop, bit di parità. L'UART è contemporaneamente un trasmettitore (TD, pin 2 nel connettore RS-232), e un ricevitore (RD pin 3 nel connettore RS-232). La figura a mostra lo schema a blocchi di un UART generico, il



quale è alimentato a una tensione +5V.

Esaminiamo ora alcuni blocchi di nostro interesse.

- **Buffer del bus dati:** isola il bus dati del terminale o computer dal bus dati interno interno dell'UART.
- **Controllo delle operazioni programmabili:** questo blocco del circuito fornisce il mezzo per la programmazione delle varie caratteristiche dell'UART e legge le informazioni di stato relative alle linee di handshaking dell'interfaccia.
- **Clock:** il pin di clock è generato esternamente. Nei sistemi digitali, i segnali di clock vengono utilizzati per sincronizzare le operazioni del sistema, ma nell'UART l'uso più importante è di temporizzatore. Il segnale di clock normalmente è pari a 16 volte il baud rate, ciò permette all'UART di sondare la linea di ricezione e di trasmettere il flusso di bit in uscita con una temporizzazione.
- **Reset:** questo input resetta l'UART, ciò accade all'accensione del computer o stampante.

Prendiamo ora in considerazione la connessione dell'UART con l'interfaccia RS-232.

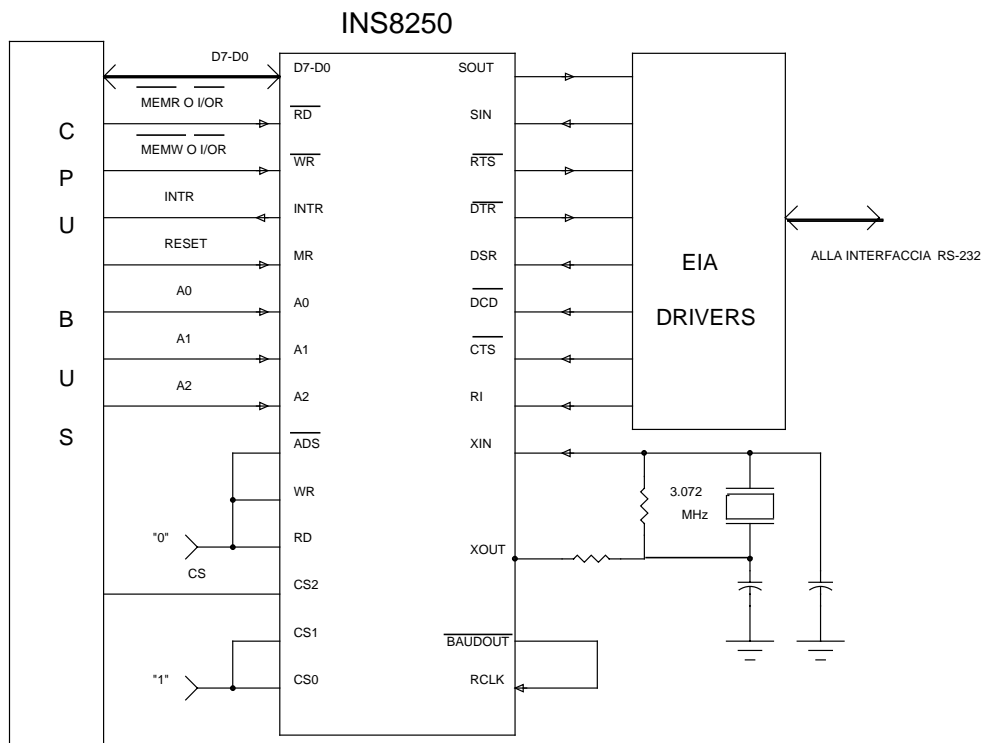
- **Controllo del trasmettitore:** questo blocco contiene il registro conservazione dati trasmessi e il shift PISO.
- **TX pronto:** è un segnale che può essere utilizzato per indicare al microprocessore che il registro trasmissione dati (hold transmit), il quale memorizza i dati di uscita trasmessi dal PISO, è vuoto ed è disponibile a ricevere un altro dato (da 8 bit).
- **Controllo del ricevitore:** questo blocco contiene il registro dati e il registro di shift SIPO.
- **RX pronto:** è l'equivalente di TX pronto, indica che il registro di ricezione dati contiene un carattere da leggere per il microprocessore.
- **Controllo di handshaking del modem:** serve per la gestione delle linee di handshaking dell'RS-232.

Il modello UART da noi utilizzato è il **national 8250** che nel seguente paragrafo verrà analizzato dettagliatamente.

Descrizione generale della UART National INS8250

L'unità 8250 è un dispositivo di interfaccia I/O programmabile denominato UART (Universal Asynchronous Receiver/Transmitter) appositamente studiato per rendere facili le comunicazioni di tipo seriale. Al suo interno è presente pure un BRG (Baud Rate Generator) generatore di baud rate che per funzionare necessita esternamente un cristallo di quarzo di frequenza opportuna. La sezione di ricezione dell'UART trasforma i dati in formato seriale provenienti da una sorgente esterna in un dato parallelo disponibile internamente nei registri interfacciati con il processore CPU del PC. La sezione di trasmissione dell'UART converte un "dato parallelo" in formato seriale aggiungendo il bit di start, il bit di parità ed i bit(s) di stop. La lunghezza della parola è programmabile a 5, 7 o 8 bits. Le possibilità di scelta per il bit di stop sono : 1, 1.5, 2.

Il generatore di "baud-rate" BRG è dotato di divisore di frequenza programmabile da 1 a $(2^{16}-1)$ per fornire le velocità standard previste dall'interfaccia RS-232C utilizzando uno dei quarzi industrialmente più diffusi (1.8432MHz, o 3.072MHz). Una uscita "bufferata" del segnale di clock denominata BAUDOUT fornisce o il segnale amplificato proveniente dal cristallo di quarzo, oppure un segnale con frequenza pari a 16 volte il clock degli shift register di ricezione e di trasmissione (uscita per usi generali) . Per interfacciare la UART ad un modem asincrono sono disponibili i segnali di controllo : RTS, CTS, DSR, DTR, RI, DCD.



Descrizione dei pin dell'integrato INS8250 (lato EIA driver)

Segnali di ingresso

- **CTS** (INIZIO INVIO DATI "CLEAR TO SEND" pin 36): quando è a livello logico basso, questo indica che il modem o il terminale, è pronto per scambiare i dati. Il segnale CTS è un ingresso di stato del modem la cui condizione può essere verificata dalla CPU e letta dal bit 4 (CTS) del registro di stato del modem. Il bit 4 è il complemento del segnale CTS esterno. Il bit 0 (DCTS) del registro di stato del modem indica se l'ingresso CTS è cambiato di stato dalla precedente lettura del registro di stato del modem. L'ingresso CTS non ha effetti sul trasmettitore.

N.B.

Tutte le volte che il bit CTS del registro di stato del modem cambia, viene generata una richiesta di interrupt, se il " Modem Status Interrupt" è stato preventivamente abilitato attraverso il bit IER3 del registro Interrupt Enable Register.

- **DSR** (TERMINALE PRONTO "DATA SET READY" pin 37): quando è a livello logico basso, questo indica che il modem o il terminale è pronto per stabilire un collegamento con l'UART. Il DSR è un ingresso di stato del modem, la cui condizione può essere verificata dalla CPU e letta dal bit 5 (DSR) del registro di stato del modem. Il bit 5 è il complemento del segnale DSR esterno. Il bit 1 (DDSR) del registro di stato del modem indica se l'ingresso DSR è cambiato di stato a partire dalla precedente lettura del registro di stato del modem.

N.B.

Tutte le volte che il bit DSR del registro di stato del modem cambia, viene generata una richiesta di interrupt, se il " Modem Status Interrupt" è stato preventivamente abilitato attraverso il bit IER3 del registro Interrupt Enable Register.

- **DCD** (RICONOSCIMENTO PORTANTE DATI "DATA CARRIER DETECT" pin 38): quando è a livello logico basso, indica che la portante dati è stata rilevata dal modem o dal terminale. Il segnale DCD è un ingresso di stato del modem la cui condizione può essere verificata dalla CPU e letta dal bit 7 (DCD) del registro di stato del modem. Il bit 7 è il complemento del ingresso DCD esterno. Il bit 3 (DDCD) del registro di stato del modem indica se l'ingresso DCD ha cambiato stato a partire dalla precedente lettura del registro di stato del modem. DCD non ha effetti sul ricevitore.

N.B.

Tutte le volte che il bit DCD del registro di stato del modem cambia, viene generata una richiesta di interrupt, se il " Modem Status Interrupt" è stato preventivamente abilitato attraverso il bit IER3 del registro Interrupt Enable Register.

- **RI** (INDICATORE DI CHIAMATA "RING INDICATOR" pin 39): quando è a livello logico basso, questo indica che uno squillo telefonico segnala che è stato ricevuto dal modem o dal terminale. Il segnale RI esterno è un ingresso di stato del modem la cui condizione può essere verificata dalla CPU e letta dal bit 6 (RI) del registro di stato del modem. Il bit 6 è il complemento del segnale RI esterno. Il bit 2 (teri) del registro di stato del modem indica se il

segnale d'ingresso RI ha cambiato da livello basso ad alto a partire dalla precedente lettura del registro di stato del modem.

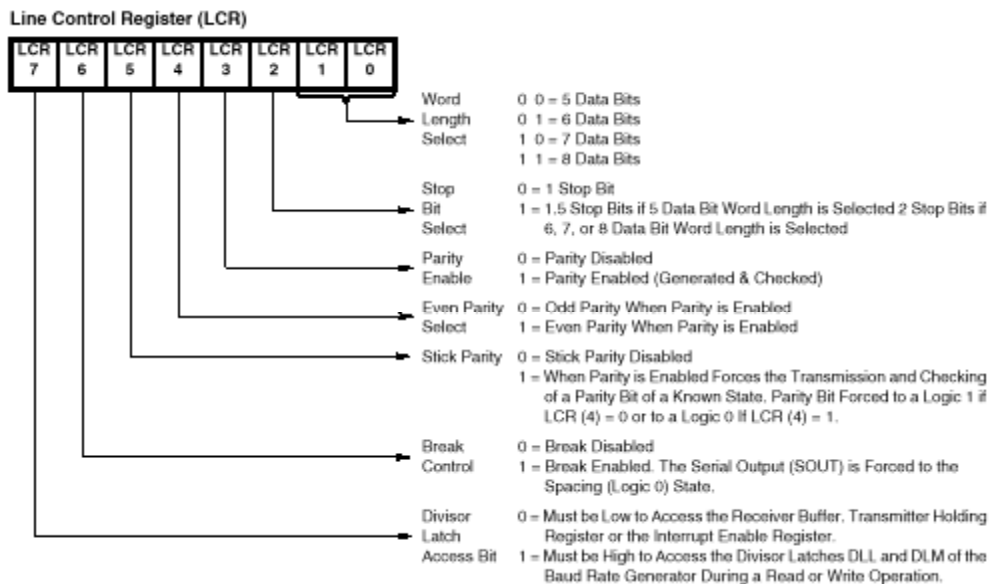
N.B.

Tutte le volte che il bit RI del registro di stato del modem cambia da alto a basso stato, viene generata una richiesta di interrupt, se il “ Modem Status Interrupt” è stato preventivamente abilitato attraverso il bit IER3 del registro Interrupt Enable Register.

Segnali di Uscita

- **DTR** (“DATA TERMINAL READY “TERMINALE PRONTO” pin 33): quando è a livello logico basso, questa uscita informa il modem o il terminale che l’unità UART è pronta per stabilire un collegamento. Il segnale di uscita DTR può essere portato a livello basso attraverso la programmazione del bit 0 (DTR) del registro di controllo del modem (scrivendo un “1”). Una condizione di “Master Reset” porta questa uscita nel suo stato logico inattivo (stato alto).
- **RTS** (RICHIESTA INVIO DATI “REQUEST TO SEND” pin 32): quando è a livello logico basso, questa uscita informa il modem o il terminale che l’unità UART è pronta per scambiare dati. Il segnale di uscita RTS può essere portato a livello basso attraverso la programmazione del bit 1 (RTS) del registro di controllo del modem (scrivendo un “1”). Una condizione di “Master Reset” porta questa uscita nel suo stato logico inattivo (stato alto).

LCR (Line Control Register/Registro delle linee di controllo)



Il programmatore specifica il formato usato dalla comunicazione seriale e fissa il “Divisor Latch bit” (DLAB) attraverso il bit di controllo di linea(LCR). Il programmatore può anche leggere il contenuto del registro LCR. La possibilità di lettura del registro LCR, semplifica la programmazione dell’UART ed elimina la necessità di dover depositare separatamente in memoria i parametri della comunicazione seriale.

I dettagli riguardanti i bit sono i seguenti :

- **BIT 0 E BIT 1:** questi due bit specificano la lunghezza della parola da trasmettere o ricevere. La codifica dei bit 0 e dei bit 1 avviene nel seguente modo.

LCR1	LCR0	Lunghezza
0	0	5 bits
0	1	6 bits

1	0	7 bits
1	1	8 bits

- **BIT 2 :** questo BIT specifica il numero di bit di stop della parola da trasmettere o da ricevere. Se il bit LCR2 è a livello logico 0, viene generato in trasmissione o controllato in ricezione un bit di stop. Se il bit LCR2 è a livello logico 1, nel caso di selezione di una parola lunga 5 bit viene generato 1.5 bit di stop in caso di selezione di parole lunghe 6, 7 o 8 bit vengono generati due bit di stop. Il ricevitore verifica sempre solo il primo bit di stop, indipendentemente dal numero di bit selezionati.
- **BIT 3 :** questo bit è il bit di abilitazione della parità. Quando il bit LCR3 è a livello logico 1, viene generato il bit di parità in trasmissione o verificato in ricezione. Tra l'ultimo bit della parola e il bit di stop dei segnali seriali. Il bit di parità è usato per rendere pari o dispari il numero complessivo di livelli "1" trasmessi .
- **BIT4:** questo bit è il bit che seleziona il tipo di parità. Quando LCR3=1 e LCR4=0, il trasmettitore e il ricevitore funzionano con parità dispari. Quando LCR3=1 e LCR4=1 il trasmettitore e il ricevitore funzionano con parità pari (trasmettitore genera, il ricevitore controlla).
- **BIT5:** questo bit è il bit di parità aggiunta. Quando il bit 3, 4 e 5 sono a livello logico 1 il bit di parità viene trasmesso e verificato come a livello logico 0. Se il bit 3 e 5 sono a livello logico 1 e il bit 4 è a livello logico 0 allora il bit di parità viene trasmesso e verificato come bit a livello logico 1. Se il bit 5 è a livello logico 0 allora il bit di parità aggiunta è disabilitato.
- **BIT 6:** questo è il bit di controllo della condizione BREAK. Esso provoca la trasmissione di una condizione BREAK dell'UART. Quando è fissato a livello logico 1, l'uscita seriale (SOUT) viene forzata in condizioni SPACE (livello logico 0). La condizione di BREAK viene disabilitata portando a livello 0 il bit 6. Il bit di controllo della condizione BREAK agisce solamente sull'uscita (SOUT) e non ha effetti sulla logica del di trasmissione..

N. B.

Questo bit permette alla CPU di allertare un terminale in un sistema di comunicazione computerizzato. Rispettando la sequenza di seguito riportata si evita la trasmissione di caratteri estranei o errati a causa di uno stato di BREAK dell'UART.

1. Carica un carattere costituito da tutti "0" in risposta alla segnalazione THRE (Registro Hold del trasmettitore vuoto) ;
2. Setta la condizione BREAK dopo il successivo THRE.
3. Attendi la condizione di inattività del trasmettitore (TSRE=1) e cancella la condizione di break quando deve essere ripristinata la normale condizione di trasmissione. Durante lo stato di BREAK, il trasmettitore può venire utilizzato come timer di carattere per stabilire con accuratezza la durata della condizione di BREAK

- **BIT 7:** questo bit è il bit di accesso per il latch di divisione. Esso deve essere fissato alto (livello logico 1) per accedere ai latch di divisione del Baud Generator durante un operazione di lettura e scrittura. Esso deve essere fissato basso (livello logico 0) per accedere al buffer di ricezione, al Transmitter Holding Register, o all'Interrupt Enable Register.

LSR (Line Status Register/Registro di stato delle linee)

LSR BITS 0 THRU 7

	LOGIC 1	LOGIC 0
LSR (0) Data Ready (DR)	Ready	Not Ready
LSR (1) Overrun Error (OE)	Error	No Error
LSR (2) Parity Error (PE)	Error	No Error
LSR (3) Framing Error (FE)	Error	No Error
LSR (4) Break Interrupt (BI)	Break	No Break
LSR (5) Transmitter Holding Register Empty (THRE)	Empty	Not Empty
LSR (6) Transmitter Empty (TEMT)	Empty	Not Empty
LSR (7) Not Used		

Il registro di 8 bit consente alla CPU di acquisire informazioni riguardo al trasferimento dei dati. Qui di seguito sono riportati i dettagli su ogni bit.

- BIT 0:** questo è il bit (DR) che indica che il ricevitore è pronto . Il bit LSR0 è a livello logico 1 ogni qual volta che un carattere entrante è stato ricevuto completamente ed è stato trasferito nel buffer di ricezione RBR. Il bit LSR0 viene portato a livello logico 0 dopo che è stata effettuata la lettura del registro RBR.
- BIT 1:** questo bit è l'indicatore di errore di overrun OE. Il bit LSR1 indica che i dati nel buffer di ricezione RBR non sono stati letti dalla CPU prima che un successivo carattere venisse trasferito all'interno del medesimo registro RBR ; questa situazione ha perciò distrutto il carattere precedentemente ricevuto. L'indicatore di errore OE viene portato a livello logico 1 quando viene rivelato sul ricevitore una condizione di errore (sovrascrittura o overrun) e resettato ogni qual volta la CPU legge il contenuto del registro LSR.
- BIT 2:** questo bit rappresenta l'indicatore di errore di parità (PARITY ERROR=PE).Bit LSR2 indica che il carattere ricevuto non è dotato di bit di parità (pari o dispari) corretto, in relazione alle condizioni di programmazione del ricevitore.
- BIT 3:** questo bit indica l'errore di struttura del dato ricevuto (FRAMING ERROR) .Questo bit indica che il carattere ricevuto non è dotato di bit di stop valido. Il bit LSR3 è portato a livello logico 1 ogni qual volta che il bit di stop che segue l'ultimo bit di dati o il bit di parità viene trovato a livello logico 0 (SPACING) .Il bit LSR3 viene resettato ogni qual volta che la CPU legge il contenuto del registro LSR. L'UART tenterà a risincronizzare il ricevitore, dopo il "framing error". Per far questo il sistema presume che il "framing error" sia dovuto al bit di start di un carattere successivo ; di conseguenza campiona questo bit 2 volte e poi inizia a interpretare ciò che segue come un dato.
- BIT 4:** questo bit è il segnalatore di BREAK INTERRUPT (BI). Il bit LSR4 è portato a livello logico 1 ogni qual volta il segnale di ingresso al ricevitore rimane in condizioni "spacing" (livello 0) per un tempo più lungo del tempo necessario per trasmettere un carattere completo di bit di start, di stop e di parità. Il bit LSR4 viene resettato ogni qual volta la CPU legge i contenuti del registro LSR. Per poter ripartire dopo la ricezione di una condizione BREAK, è necessario che l'ingresso seriale del ricevitore SIN si trovi a livello logico 1 per almeno il tempo corrispondente alla durata di 1/2 bit.

N. B.

I bit da 1 a 4 rappresentano le situazioni di errore che producono una condizione di Line Status Interrupt ognivolta che una determinata condizione è presente e l'interrupt viene abilitato.

- **BIT 5:** questo bit è l'indicatore (THRE) "registro holding di trasmissione vuoto". Il bit 5 indica che l'UART è pronta per accettare un nuovo carattere per la trasmissione. In aggiunta questo bit fa in modo che l'UART mandi un'interruzione alla CPU quando il bit THREI (abilitazione dell'interrupt THRE) è settato a livello logico 1. Il bit THRE passa a livello logico 1 quando un carattere viene trasferito dal THR al TSR (transmitter shift register). Il bit passa a livello logico 0 ogni qual volta che la CPU carica un dato all'interno del registro THR.
- **BIT 6:** questo bit mi indica che il registro TSR (Transmitting Shift Register) è vuoto. Il bit 6 passa a livello logico 1 ogni qual volta che il TSR è vuoto; il bit 6 passa a livello logico 0 ogni qual volta un dato viene trasferito al registro TSR.
- **BIT 7:** questo bit resta permanentemente fissato a livello logico 0.

N.B. Sul registro LSR è possibile effettuare solo operazioni di lettura ;è sconsigliato effettuare operazioni di scrittura, poiché esse vengono effettuate solamente per ragioni di collaudo nelle fabbriche dove il circuito integrato viene costruito.

MSR (Line Modem Register)

MSR BITS 0 THRU 7

MSR BIT	MNEMONIC	DESCRIPTION
MSR (1)	DDSR	Delta Data Set Ready
MSR (2)	TERI	Trailing Edge of Ring Indicator
MSR (0)	DCTS	Delta Clear To Send
MSR (3)	DDCD	Delta Data Carrier Detect
MSR (4)	CTS	Clear To Send
MSR (5)	DSR	Data Set Ready
MSR (6)	RI	Ring Indicator
MSR (7)	DCD	Data Carrier Detect

Questo registro fornisce informazioni alla CPU sullo stato attuale delle linee di controllo del modem (o di un altro dispositivo periferico). Quattro bit del registro MSR consentono pure di segnalare i cambiamenti di stato sulle linee di controllo ; questi bit vengono portati a livello logico 1 ogni qual volta un ingresso di controllo del modem cambia di stato . Essi vengono resettati a 0 ogni qual volta che la CPU legge il registro MSR.. I dettagli su ogni bit dell'MSR sono riportati di seguito.

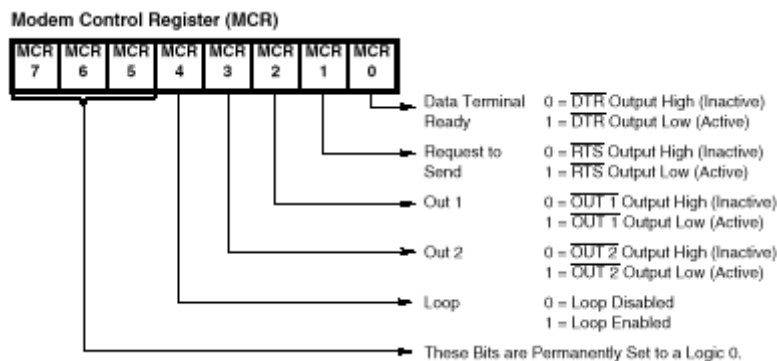
- **BIT 0:** questo bit è l'indicatore "Delta Clear To Send" (DCTS). Il bit MSR0 indica che l'ingresso CTS ha cambiato stato a partire dall'ultima volta che è stato letto dalla CPU.
- **BIT1:** questo bit è l'indicatore "Delta Data Set Ready" (DDSR). Il bit MSR1 indica che l'ingresso DSR ha cambiato stato a partire dall'ultima volta che è stato letto dalla CPU.
- **BIT 2:** questo bit è l'indicatore rilevatore "Trailing Edge of Ring Indicator" (TERI). Il bit MSR2 indica che l'ingresso RI ha cambiato stato; è passato dal livello logico basso al livello logico alto.
- **BIT 3:** questo bit è l'indicatore "Delta Data Carrier Detect" (DDCD). Il bit MSR3 indica che l'ingresso DCD ha cambiato stato.

N.B.

Ogni qual volta che i bit 0, 1, 2 e 3 si trovano i a livello logico 1, viene generato un modem status interrupt.

- **BIT 4:** questo bit è il complemento della linea di ingresso “Clear to send” (CTS); se il bit MCR4 (loop) dell’MCR è a livello logico 1, questo bit è l’equivalente all’RTS nell’MCR.
- **BIT 5:** questo bit è il complemento della linea di ingresso “Data Set Ready (DSR)); se il bit MCR4 (registro MCR) è a livello logico 1, questo bit è equivalente al DTR nel registro MCR.
- **BIT 6:** questo bit è il complemento della linea di ingresso Ring Indicator (RI); se il bit MCR4 dell’MCR è a livello logico 1, questo bit è equivalente alla linea OUT1 nell’MCR.
- **BIT 7:** questo bit è il complemento della linea di ingresso “Data Carrier Detect” (DCD); se il bit MCR4 dell’MCR è a livello logico 1, questo bit è equivalente alla linea OUT2 nell’MCR.

MCR (Modem Control Register)



Il registro di controllo MCR è l’interfaccia verso il modem o il “data set” (o dispositivo periferico simile a un modem). Passiamo ora all’esame dettagliato dei vari bit che questo registro contiene.

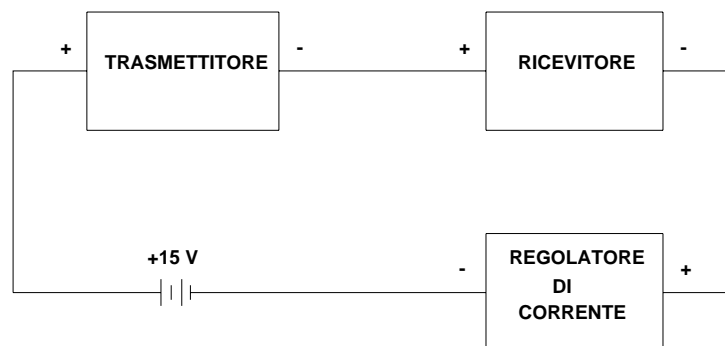
- **BIT 0:** Questo bit controlla l’uscita DTR “Data Terminal Ready”; quando il bit MCR0 è a livello logico 1, l’uscita DTR si porta a livello 0; quando il bit MCR0 è a livello logico 0, l’uscita DTR si porta a livello 1.
- **BIT 1:** Questo bit controlla l’uscita RTS “Request to Send”. Il bit MCR1 ha effetti sull’uscita RTS identici a quelli descritti per il bit MCR0.
- **BIT 2:** Questo bit controlla l’uscita OUT1 che è un’uscita aggiuntiva di controllo. Il bit MCR2 ha effetti sull’uscita OUT1 identici a quelli descritti per il bit MCR0.
- **BIT 3:** Questo bit controlla l’uscita OUT2 che è un’uscita aggiuntiva di controllo. Il bit MCR3 ha effetti sull’uscita OUT2 identici a quelli descritti per il bit MCR0.
- **BIT 4:** questo bit fornisce una “loopback locale” per test diagnostici da effettuare sulla UART. Quando il bit MCR4 è portato a livello 1 succede quanto segue: l’uscita del trasmettitore seriale SOUT viene mantenuta in stato MARK (livello logico 1); l’entrata del ricevitore seriale SIN viene sconnessa; l’uscita del TST (Trasmitter Shift Register) viene internamente connessa con l’ingresso dello SR di ricezione. I quattro ingressi di controllo del modem (CTR, DSR, RI e DCD) vengono sconnessi; le quattro uscite di controllo del modem (DTR, RTS, OUT1 e OUT2) vengono internamente connesse ai quattro ingressi di controllo del modem. In “modalità diagnostica” il dato che viene trasmesso viene immediatamente ricevuto. Questa caratteristica permette al processore di verificare i percorsi di trasmissione e di ricezione dei dati all’interno dell’UART. In “modalità diagnostica” gli interrupt di trasmissione e di ricezione sono perfettamente operativi. Gli Interrupt del registro MCR sono anch’essi operativi, mentre le

sorgenti di interrupt sono ora i 4 bit bassi dell'MCR invece delle quattro entrate di controllo del modem. Le interruzioni sono ancora controllati dal registro di abilitazione degli interrupt (IER).

- **BIT DAL 5 AL 7:** questi bit sono permanentemente tenuti a livello 0.

Interfaccia “Current Loop” (Loop di Corrente)

Questo tipo di interfaccia, inizialmente adottato nei sistemi telegrafici viene utilizzato per la trasmissione di dati seriali asincroni, ed ha il grande vantaggio di una notevole semplicità e versatilità . Purtroppo i suoi parametri caratteristici non sono standardizzati, e possono sorgere difficoltà per il progettista . La figura sottostante mostra lo schema di principio di un collegamento in “current loop” a 20 mA:



I livelli di tensione ai quali sono abbinate le informazioni binarie 0 e 1 vengono sostituiti dalla presenza o assenza di corrente (cioè livello logico 1 coincide con la presenza di corrente, mentre l'assenza di quest'ultima individua livello logico 0). Il valore 20 mA è stato scelto inizialmente poiché era il numero che garantiva un contatto elettrico sicuro nei dispositivi elettromeccanici. Attualmente questi elementi sono stati sostituiti da dispositivi a stato solido quali i disaccoppiatori optoelettronici . Dato che lavoriamo in loop possiamo posizionare il ricevitore ad una distanza di 850 m dal trasmettitore (sempre che non ci siano disturbi o altre mragioni che lo impediscano). Nei circuiti telegrafici con distanze superiori si utilizzano tensioni elevate (120 o 160 V) e dispositivi con resistenze interne più piccole per permettere lunghezze superiori dei cavi di collegamento. Il generatore di corrente può essere localizzato nel trasmettitore o nel ricevitore e viene chiamato attivo il dispositivo che lo contiene, mentre passivo l'altro. Il collegamento diretto può dunque realizzarsi solamente fra un terminale attivo ed uno passivo, e non fra due attivi o due entrambi passivi. Questo tipo di circuito ha inoltre il vantaggio di poter funzionare con più trasmettitori e ricevitori collegati, anche se deve valere sempre la condizione che ci sia un solo dispositivo attivo.

Stampante Epson LX- 800

L'interfaccia seriale 8148D presente nella stampante è un dispositivo che consente il collegamento della nostra stampante EPSON LX-800 a un computer attraverso una linea di comunicazione asincrona (RS-232C). Le caratteristiche essenziali sono :

- *Velocità di trasmissione tra 75 e 19200 baud, con limitazione a 1200 baud se si utilizza il Current Loop a 20 mA.*
- *Un buffer da 2 KB che sveltisce le operazioni di stampa liberando il computer dai tempi di attesa.*
- *Protocollo di handshaking XON/XOFF e flag DTR.*
- *Due funzioni self test per la diagnostica dell'interfaccia (Loopback o Line monitor).*

L'interfaccia seriale 8148D contiene due banchi di DIP switch (SW1 e SW2) ed un interruttore esterno di attivazione.

Interruttore di attivazione

L'interruttore di attivazione, permette di attivare e disattivare l'interfaccia senza doverla smontare fisicamente dalla stampante. In posizione ON (verso l'alto) è attivata l'interfaccia seriale e disattivata quella parallela ; in posizione OFF (verso il basso) è attivata l'interfaccia parallela e disattivata quella seriale.

Dip Switch

Tabella SW1

DIP SWITCH FUNZIONE	ON	OFF	DEFAUL
1-1 bit di dati	7	8	OFF
1-2 parità	ON	OFF	OFF
1-3 tipo di parità	Pari	Dispari	OFF
1-4 polarità	Negativa	Positiva	OFF

I DIP switch 1-5, 1-6, 1-7, 1-8 ci servono per stabilire la velocità di trasmissione in base alle combinazioni degli stessi DIP switch come rappresentato in tabella 1.

Tabella SW2

Dip Switch	Funzione	ON	OFF	Default
2 - 1	Non usato	-----	-----	Off
2 - 2	Buffer	On	Off	On
2 - 3	Ripristino trasmissione	Vedi tabella 2	Vedi tabella 2	Off
2 - 4	Ripristino trasmissione	Vedi tabella 2	Vedi tabella 2	Off
2 - 5	Self - test	On	Off	Off
2 - 6	Tipo self - test	Line monitor	Loop back	Off

Per fissare i DIP switch 2-3 e 2-4 dovremo osservare la tabella 2, dalla quale ricaveremo il nostro spazio libero nel buffer in byte.

Velocita' di trasmissione

Qualunque combinazione differente da una di quelle riportate sotto imposta una velocità di 19200 baud.

Tabella 1 Velocità di trasmissione

Baud	Dip Switch			
	1 - 5	1 - 6	1- 7	1 - 6
75	On	On	On	On
100	On	On	On	Off
134.5	On	On	Off	On
150	On	On	Off	Off
200	On	Off	On	On
300	On	Off	On	Off
600	On	Off	Off	On
1200	On	Off	Off	Off
1800	Off	On	On	On
2400	Off	On	On	Off
4800	Off	On	Off	On
9600	Off	On	Off	Off
19200	Off	Off	On	On

Quando lo spazio libero nel buffer di input scende a meno di 80 byte, la ricezione dati viene interrotta. Naturalmente, mano a mano che la stampante svuota il buffer si viene a liberare dello spazio. Quando lo spazio libero raggiunge uno dei valori indicati nella tabella 2, la trasmissione viene ripristinata. I DIP switch 2-3 e 2-4 controllano questa funzione.

Tabella 2

Spazio libero nel buffer Byte	Dip Switch	
	2 - 3	2 - 4
152	Off	Off
288	Off	On
560	On	Off
1936	On	On

Connettore e Segnali

Il connettore dell'interfaccia è uno standard CANNON a 25 PIN. La tabella 3 descrive i pin dei segnali sul connettore dell'interfaccia.

Tabella 3

Pin	Segnale	I / O	Descrizione
1	Massa di sistema	-----	Massa a terra dello chassis
2	TXD (transmitted data)	Out	Trasmissione dati
3	RXD (recived dati)	In	Ricezione dati
6	DSR (data set ready)	In	Deve essere a livello basso perchè la stampante possa ricevere dati
7	Massa del segnale	-----	Massa del segnale
8	DCD (data carrier in detect)	In	Equivalente a DSR pin 6
11	RTS (reserv chanel)	Out	Deve essere a livello basso perchè la stampante possa ricevere dati
17	TTY - TXD	Out	Una bassa impedenza tra i punti 17 e 24 o uno X on sugli stessi pin
20	DTR (data terminal out ready)	Out	Deve essere a livello basso perchè la stampante possa ricevere dati
23	TTY - RXD return	-----	-----
24	TTY - TXD return	-----	Indica che la stampante è pronta a ricevere dati
25	TTY - RXD	In	Ricezione dati in current loop

La direzione IN o OUT è riferita rispetto alla stampante. L'utente può modificare la polarità del segnale attraverso il DIP switch 1-4.

Trasmissione Dati

L'interfaccia può accettare i dati dal calcolatore quando il segnale di DCD o il segnale di DSR è ALTO. In altre parole, questa interfaccia funziona da UART (Universal Asynchronous Receiver/Transmitters).

I dati ricevuti dal calcolatore sono trasferiti sequenzialmente dall'interfaccia alla stampante senza alcuna trasformazione.

*Se esiste un errore (parità, velocità, ecc.) nei dati trasmessi, verrà stampato un asterisco (*) nel punto in cui è stato rilevato l'errore stesso.*

Il Buffer

L'interfaccia 8148D è dotata di un buffer. Quando il buffer si satura, la trasmissione viene sospesa e non viene riattivata fino a quando una certa quantità di spazio nel buffer non è di nuovo libera per i dati da stampare. Anche se il DIP switch 2-2 è in posizione OFF e cioè il buffer è disabilitato, la trasmissione è sospesa fino a quando i dati già acquisiti non sono stampati. Questa interfaccia è dotata di un protocollo XON/XOFF, il quale gestisce il flusso di segnali verso la stampante controllando un flag. Il calcolatore deve riconoscere lo stato del flag prima di inviare nuovi dati.

Ci sono molti sistemi per controllare la trasmissione seriale dei dati che, in genere, utilizzano i pin CTS, DSR..

Buffer disabilitato

In questo caso, il protocollo XON/XOFF non è utilizzabile e il solo controllo possibile è attraverso un flag. Tale flag è costituito dallo stato del pin 11 (RTS reverse channel) in RS-232C, mentre in Current Loop è costituito dallo stato del pin 17 (TTY - TXD). Tale trasmissione può essere attivata o disattivata dallo stato di uno dei due pin (a seconda del protocollo usato) : il significato del flag può essere invertito usando il DIP switch 1-4, lo switch di controllo della polarità del segnale.

Buffer abilitato

Quando tale è abilitato, il controllo della trasmissione dati può essere gestito per mezzo del protocollo XON/XOFF.

Controllo tramite flag

La massima capacità del buffer è di 2 KB. Quando la velocità di trasmissione è più alta di quella di stampa, il buffer si riempie fino a quando non restano che 16 byte liberi. A quel punto, l'interfaccia modifica lo stato del flag, disattivando la trasmissione. I segnali utilizzati come flag sono gli stessi visti sopra : 20 (DTR) in RS-232C e 17 (TTY - TXD) in Current Loop. Una volta bloccata la trasmissione dati, il buffer inizia a svuotarsi, mano a mano che i dati vengono stampati. Quando l'area libera raggiunge uno dei valori prestabiliti in tabella 2, lo stato del flag viene invertito e la trasmissione può riprendere.

Controllo tramite protocollo XON/XOFF

Il meccanismo di controllo della capacità del buffer agisce in maniera analoga a quella vista in precedenza. Mano a mano che il buffer si riempie e raggiunge un valore di soglia prestabilito (16 e 8 byte), la stampante invia un XOFF (13 hex) che sospende la trasmissione. Quando il buffer risulta sufficientemente capiente, viene trasmesso un XON (11 hex) che riabilita la trasmissione. Il segnale XON è trasmesso a intervalli regolari per confermare al PC la disponibilità della stampante a ricevere i dati. Nella tabella 4 sono riportati i tempi di XON/XOFF.

Tabella 4

VELOCITA'	XON/XOFF
75	1.06 sec
110	0.72 sec
134.5	0.59 sec
150	0.53 sec
200	0.40 sec
300	0.26 sec
600	0.13 sec
1200	66.0 msec
1800	44.0 msec
2400	33.0 msec
4800	16.5 msec
9600	8.3 msec
19200	4.1 msec

I dati trasmessi dal PC alla stampante dopo l'invio di un segnale XOFF possono andare perduti se eccedono lo spazio ancora libero del buffer. La frequenza di trasmissione dei segnali XON/XOFF è equivalente a 80 volte il tempo di trasmissione di un bit, ovvero, con una lunghezza di parola di 10 bit, a circa 8 caratteri.

Test

L'interfaccia dispone di due modi di test che possono essere selezionati agendo su due DIP switch.

Tabella 5

2 - 5	2 - 6	Self - test
On	Off	Loop back
On	On	Line monitor

Loopback

All'accensione della stampante con i pin 2 (TXD) e 3 (RXD) connessi tra di loro, vengono trasmessi e stampati caratteri tra 20 hex e 7E hex (da 32 a 126 decimale). L'interfaccia deve essere in modo RS-232C : in Current Loop questa operazione non può essere effettuata.

Line monitor

In questo caso la stampante si comporta come nel modo Hex Dump di cui dispongono diverse serie di stampanti. I dati ricevuti vengono stampati in esadecimale invece che nel corrispondente carattere ASCII.

Programmazione Stampante Epson Lx-800

Per effettuare l'impostazione della nostra stampante , dovremo impostare per prima cosa fissare le funzioni dei due banchi di DIP switch contenuti nell'interfaccia seriale 8148D.

Per semplificare ogni spiegazione riporteremo gli schemi dei due banchi di DIP switch con le relative impostazioni :

Tabella SW1

Funzione degli switch del banco 1

Dip switch	Funzione	Impostazione
1-1	Bit di dati	8
1-2	Parità	OFF
1-3	Tipo parità	Dispari
1-4	Polarità	Positiva
1-5	Vel.Trasmissione	ON/300 Baud
1-6	Vel.Trasmissione	OFF/300 Baud
1-7	Vel.Trasmissione	ON/300 Baud
1-8	Vel.Trasmissione	OFF/300 Baud

Per impostare i DIP switch 1-5, 1-6 , 1-7, 1-8, bisogna osservare la tabella 1, e con le combinazioni ON/OFF possiamo ottenere la velocità di trasmissione della nostra stampante, nel nostro caso è di 300 BAUD.

Tabella SW2

Funzione degli switch del banco 2

DIP switch	Funzione	Impostazione
2-1	Non usato	Non usato
2-2	Buffer	ON
2-3	Ripristino Trasm.	OFF/152 byte
2-4	Ripristino Trasm	OFF/152 byte
2-5	Self-Test	OFF
2-6	Tipo Self-Test	Loopback

Per impostare i DIP switch 2-3, 2-4, bisogna osservare la tabella 2, e con le combinazioni ON/OFF possiamo ottenere il valore dello spazio libero nel buffer, nel nostro caso è di 152 byte.

Programmazione in Turbo Pascal 6.0

Questo linguaggio venne creato da Niklaus Wirth all'inizio degli anni '70 per insegnare a programmare . Perciò è adattissimo come primo linguaggio di programmazione, mentre chi ha già programmato con altri linguaggi, lo troverà facile da apprendere.

Per iniziare parleremo dei concetti basilari del linguaggio Pascal indicandone gli elementi base della programmazione.

Il compito del programmatore consiste nel :

- *Introdurre le informazioni nel programma - **input***
- *Creare uno spazio in cui memorizzarle - **dati***
- *Usare le istruzioni corrette 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 (Read, Readln, Port)

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 (Write, Writeln, Port)

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 (2.579x10E24). 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, conterranno delle informazioni.

Per spiegare meglio i tipi di dati interi e reali possiamo rappresentarli in due tabelle, che conterranno 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 x 10E-39 .. 1.7 x 10E38	11 - 12	6
Single	1.5 x 10E-45 .. 3.4 x 10E38	7 - 8	4
Double	5.0 x E10-324 .. 1.7 x 10E308	15 - 16	8
Extended	1.9 x 10E-4951 .. 1.1 x 10E4932	19 - 20	10
Comp*	-2E+63+1 .. 2E+63-1	19 - 20	8

*Comp può contenere solo valori interi.

Libreria di Gestione della UART 8250 in Turbo Pascal 6.0

Questa libreria contiene dei programmi che ci servono per poter ricevere i dati che provengono dal centralino. La libreria prevede una procedura di inizializzazione che va chiamata prima di eseguire qualsiasi operazione sulla seriale. Questa inizializzazione consiste nel fissaggio dei dati seriali, in modo che coincidano con quelli provenienti dal centralino.

Nel nostro caso per inizializzare la nostra porta seriale COM1 con i valori 300,N,8,2 dovremo introdurre nel codice la riga :

```
Init_Com ('COM1',300,'N',8,2) ;
```

Dove la COM1 è la nostra porta seriale, 300 è la velocità di trasmissione dei dati, N vuol dire no parità, 8 è il numero di bit di dati e 2 è il numero di bit di stop. Dopo di questo si potranno chiamare le altre procedure ,se saranno necessarie ,tra le seguenti :

- **MI_Status : Modem line control Status:** segnala sul video se esistono linee di controllo del modem che sono in stato di inattività (high).
- **Rx_Status : Receiver Status:** segnala sul video se esistono problemi sul ricevitore (Overrun, Framing Error , Break).
- **TX_Ch (D : char) :** Trasmette un carattere sulla seriale inizializzata.
- **DTR_ON :** Attiva la linea di controllo DTR , porta a livello 0 DTR : uscita della seriale.
- **DTR_OFF :** Disattiva la linea di controllo DTR , porta a livello 1 DTR : uscita della seriale.
- **RTS_ON :** Attiva la linea di controllo RTS , porta a livello 0 RTS : uscita della seriale.
- **RTS_OFF :** Disattiva la linea di controllo RTS , porta a livello 1 RTS : uscita della seriale.
- **DSR : boolean :** Restituisce TRUE se la linea di controllo DSR ingresso della seriale è attiva (a livello 0) FALSE in caso contrario.

- **CTS : boolean** : Restituisce *TRUE* se la linea di controllo CTS ingresso della seriale è attiva (a livello 0) *FALSE* in caso contrario.
- **DCD : boolean** : Restituisce *TRUE* se la linea di controllo DCD ingresso della seriale è attiva (a livello 0) *FALSE* in caso contrario.
- **RX_Ch : CHAR** : Riceve un carattere dalla seriale.

{ Libreria di gestione della Seriale UART 8250 NSC

La libreria prevede una procedura di inizializzazione che va chiamata prima di effettuare qualsiasi operazione sulla seriale. Ad esempio, volendo inizializzare COM2 a 4800,N,8,1 si dovr... introdurre nel codice la riga:

```
Init_Com('COM2',4800,'N',8,1);
```

dopodiche' si potranno chiamare sia le altre procedure RX_CH, TX_CH, MI_Stat, RX_stat, ecc. ecc.

Documentazione consultata Data sheet UART 8250 National Semiconductor

Tested 30-01-97 by Cleto Azzani

Revisione 23 maggio 97

}

Unit Ser_8250;

INTERFACE

Uses Bios;

Const

```
ComAddr: word = $03F8 ; { Default = COM1 }
```

{ Offset registri Interni UART 8250 }

```
IER = 1; { }  
LCR = 3; { }  
MCR = 4; { }  
LSR = 5; { }  
MSR = 6; { }
```

Type

```
str4 = string[4];
```

Procedure Init_Com(C:str4; BPS: integer; P:char; Bit:byte;Stop:byte);

{ C : COM1 o COM2 (stringa 4 caratteri lettere solo maiuscole)
BPS : velocit... di trasmissione (valori permessi 9600-4800-2400-1200-600)
P : carattere maiuscolo parit... (N no parity E even parity O odd parity)
Bit : numero bit (5 - 6 - 7 - 8)
Stop : numero bit di stop (1 - 2)

Il codice speed di inizializzazione determina la velocita' di trasmissione

ROLFI, FALETTI, STANGA

con riferimento alla seguente tabella:

Baud Rate	Valore speed
9600	12
4800	24
2400	48
1200	96
600	192
300	384

```
Procedure MI_Status; { Modem line control Status
    Segnala sul video se esistono linee di controllo
    del modem che sono in stato di inattivita'(high) }

Procedure RX_Status; { Receiver Status
    Segnala sul video se esistono problemi sul
    ricevitore : Overrun, Framing Error, Break }

Procedure DTR_ON; { Attiva la linea di controllo DTR
    porta a livello 0 DTR : uscita della seriale }

Procedure DTR_OFF; { Disattiva la linea di controllo DTR
    porta a livello 1 DTR : uscita della seriale }

Procedure RTS_ON; { Attiva la linea di controllo RTS
    porta a livello 0 RTS : uscita della seriale }

Procedure RTS_OFF; { Disattiva la linea di controllo RTS
    porta a livello 1 RTS : uscita della seriale }

Function DSR : boolean ; { Restituisce True se la linea di controllo DSR
    ingresso della seriale e' attiva (a livello 0)
    False in caso contrario }

Function CTS : boolean ; { Restituisce True se la linea di controllo CTS
    ingresso della seriale e' attiva (a livello 0)
    False in caso contrario }

Function DCD : boolean ; { Restituisce True se la linea di controllo DCD
    ingresso della seriale e' attiva (a livello 0)
    False in caso contrario }

Function TX_Ready: Boolean; { Restituisce True se TX e' pronto per trasmettere
    False in caso contrario }

Procedure TX_Ch(D : Char) ; { Trasmette un carattere sulla seriale
    inizializzata }

Function RX_Ready: Boolean; { Restituisce True se RX e' pronto per ricevere
    False in caso contrario }

Function RX_Ch:CHAR;      { Riceve un carattere dalla seriale }
```

IMPLEMENTATION

```
Procedure Init_Com(C:str4; BPS: integer; P:char; Bit:byte;Stop:byte);
{ C : COM1 o COM2 (stringa 4 caratteri lettere solo maiuscole)
  BPS : velocit... di trasmissione (valori permessi 9600-4800-2400-1200-600)
  P : carattere maiuscolo parit... (N no parity E even parity O odd parity)
  Bit : numero bit (5 - 6 - 7 - 8)
  Stop : numero bit di stop (1 - 2)
```

Il codice speed di inizializzazione determina la velocit... di trasmissione con riferimento alla seguente tabella:

Baud Rate	Valore speed
9600	12
4800	24
2400	48
1200	96
600	192
300	384

}

Var

speed : integer;
code : byte;

BEGIN

If C='COM2' Then ComAddr:= \$02F8

Else

Begin

If C<>'COM1' Then { Default = COM1 \$03F8 }

Begin

Writeln('COMx Error ',C);

Halt;

End;

End;

Case BPS of

19200 : speed := 6;

9600 : speed := 12;

4800 : speed := 24;

2400 : speed := 48;

1200 : speed := 96;

600 : speed := 192;

300 : speed := 384;

Else

Begin

Writeln('BPS Error ',BPS);

Halt;

End;

End;

code := 0;

Case P of

'N' : ;

'E' : code := code + \$18;

'O' : code := code + \$08;

Else

Begin

Writeln('P Error ',P);

Halt;

End;

End;

Case Bit Of

5 : ;

6 : code := code + \$01;

7 : code := code + \$02;

8 : code := code + \$03;

Else

Begin

Writeln('Bit Error ',Bit);

```

    Halt;
  End;
End;

```

```

Case Stop Of

```

```

  1 : ;
  2 : code := code + $04;
  Else
    Begin
      Writeln('Stop Error ',Stop);
      Halt;
    End;
End;

```

```

PORT[ComAddr+LCR] := $80 ;
PORT[ComAddr] := Lo(speed) ;
PORT[ComAddr+1] := Hi(speed) ;
PORT[ComAddr+LCR] := code ; { Attivazione formato RX TX }
PORT[ComAddr+MCR] := 0 ;
PORT[ComAddr+IER] := 0 ; { Disattivazione degli Interrupt da seriale }
End;

```

```

Procedure MI_Status; { Modem line control Status }

```

```

  Var
    a : byte;
  Begin
    a:=Port[ComAddr+MSR];
    { Writeln(' MSR ',Bin(a)); }
    If (a AND $80)=0 Then Writeln('DCD line high ');
    If (a AND $40)=0 Then Writeln('RI line high ');
    If (a AND $20)=0 Then Writeln('DSR line high ');
    If (a AND $10)=0 Then Writeln('CTS line high ');
  End;

```

```

Procedure RX_Status; { Receiver Status }

```

```

  Var
    a : byte;
  Begin
    a:=Port[ComAddr+LSR];
    { Writeln(' LSR ',Bin(a)); }
    If (a AND $02)<>0 Then Writeln('Overrun Error ');
    If (a AND $04)<>0 Then Writeln('Parity Error ');
    If (a AND $08)<>0 Then Writeln('Framing Error ');
    If (a AND $10)<>0 Then Writeln('Break ');
  End;

```

```

Procedure DTR_ON; { Attiva la linea di controllo DTR }

```

```

  Var a : byte;
  Begin
    a := PORT[ComAddr+MCR];
    PORT[ComAddr+MCR]:= a OR $01 ;
  End;

```

```

Procedure DTR_OFF; { Disattiva la linea di controllo DTR }

```

```

  Var a : byte;
  Begin
    a := PORT[ComAddr+MCR];
    PORT[ComAddr+MCR]:= (A AND $F7) ;
  End;

```

```

Procedure RTS_ON; { Attiva la linea di controllo RTS }

```

ROLFI, FALETTI, STANGA

```
Var a : byte;
Begin
  a := PORT[ComAddr+MCR];
  PORT[ComAddr+MCR]:= a OR $02 ;
End;
```

```
Procedure RTS_OFF; { Disattiva la linea di controllo RTS }
  Var a : byte;
Begin
  a := PORT[ComAddr+MCR];
  PORT[ComAddr+MCR]:= (A AND $FD) ;
End;
```

```
Function DSR : boolean ; { Restituisce True se il controllo DSR Ğ attivo
  False in caso contrario }
Begin
  If (PORT[ComAddr+MSR] AND $20) <> 0
  Then DSR := True
  Else DSR := False;
End;
```

```
Function CTS : boolean ; { Restituisce True se il controllo CTS Ğ attivo
  False in caso contrario }
Begin
  If (PORT[ComAddr+MSR] AND $10) <> 0
  Then CTS := True
  Else CTS := False;
End;
```

```
Function DCD : boolean ; { Restituisce True se il controllo DCD Ğ attivo
  False in caso contrario }
Begin
  If (PORT[ComAddr+MSR] AND $80) <> 0
  Then DCD := True
  Else DCD := False;
End;
```

```
Function TX_Ready: Boolean; { Restituisce True se TX Ğ pronto per trasmettere
  False in caso contrario }
Begin
  If ((PORT[ComAddr+LSR] AND $20) = $20)
  Then TX_ready := True
  Else TX_ready := False;
End;
```

```
Procedure TX_Ch(D : Char) ; { Trasmette un carattere }
Begin
  While (PORT[ComAddr+LSR] AND $20) <> $20 DO;
  PORT[ComAddr] := ORD(D);
End;
```

```
Function RX_Ready: Boolean; { Restituisce True se RX Ğ pronto per ricevere
  False in caso contrario }
Begin
  If ((PORT[ComAddr+LSR] AND $01) = $01 )
  Then RX_ready := True
  Else RX_ready := False;
End;
```

```
Function RX_Ch:CHAR; { Riceve un carattere }
Begin
  While (PORT[ComAddr+LSR] AND $01) <> $01 DO;
```

ROLFI, FALETTI, STANGA

```
    RX_Ch := Chr(Port[ComAddr]);  
End;
```

```
End. { Unit Ser_8250 }
```