Il livello di linea o livello di data link

Costituisce il secondo strato del modello OSI e definisce le regole attraverso le quali avviene il dialogo fra due nodi di una rete. Esso si trova fra il livello fisico, il cui scopo è quello di garantire il corretto trasferimento di bit lungo un canale, ed il livello di rete che si occupa di gestire l'instradamento dei dati all'interno di una rete di computer.

Nel livello di linea i dati vengono organizzati in frame (tipicamente poche centinaia di byte) che vengono trasmessi sequenzialmente. Tale livello si occupa delle regole

per l'instaurarsi del dialogo fra trasmettitore e ricevitore, della loro sincronizzazione, delle regole per l'interruzione della comunicazione, della verifica della presenza di errori nella ricezione del frame utilizzando caratteri di controllo, della ritrasmissione degli stessi frame quando si rileva un errore.

Servizi offerti

I sevizi offerti da tale livello si possono raggruppare in tre categorie di crescente complessità:

- servizi non orientati alla connessione e senza riscontro: sono i più semplici in assoluto. Potremmo dire che ignorano tutti i problemi possibili. Si dicono non orientati alla trasmissione poiché il trasmettitore non si preoccupa di stabilire una connessione con il ricevitore, cioè inizia la trasmissione dei frame senza assicurarsi che il ricevitore sia presente. Sono detti senza riscontro poiché non è previsto che il ricevitore invii un messaggio di conferma dell'avvenuta ricezione dei dati;
- servizi non orientati alla connessione con riscontro. Come nel caso precedente il trasmettitore non verifica la presenza del ricevitore ma si richiede che quest'ultimo invii un messaggio di conferma ogni volta che riceve un frame (messaggio detto di acknoledged ACK).
- Servizi orientati alla connessione. In questo caso il trasmettitore, prima di inviare i frame, stabilisce o cerca di stabilire una comunicazione logica con il ricevitore inviandogli segnali di richiesta (Enquiry) a cui deve ricevere risposte di assenso del tipo ACK.

Struttura dei pacchetti

Per prima cosa lo strato di linea riceve il pacchetto dati dal livello di rete e lo racchiude in un frame. Il frame è costituito dal pacchetto ricevuto dallo strato superiore con l'aggiunta di caratteri di controllo. Il formato del frame può essere di diversi tipi.

UN primo tipo è il frame a conteggio di byte: il formato è il seguente

Lunghezza in byte del frame	**Campo dati**	**Byte di checksum**

Il primo byte contiene il numero di byte che compongono il frame. Seguono i byte che compongono il campo dati ed infine vi è uno o più byte di checksum. Il byte di checksum è un byte determinato con un algoritmo particolare in base ai byte del campo dati. Un esempio di algoritmo è il seguente: il checksum viene posto dapprima a zero, poi se ne fa la OR esclusiva con il primo byte del campo dati, del risultato si fa la OR esclusiva con il secondo byte e così via. In ricezione viene eseguito di nuovo il calcolo e si fa il confronto fra il checksum ricavato dal ricevitore e quello calcolato dal trasmettitore e inviato con il frame. Se differiscono vuol dire che vi è stato qualche errore nella trasmissione del frame (vedi esempio nel foglio excel Checksum).

In definitiva con questo tipo di formato, il ricevitore può effettuare due tipi di controlli: un controllo sul numero di byte ricevuti ed un controllo sulla correttezza dei dati. Questo tipo di formato, benché semplice, presenta, però, un problema. Se,

infatti, si ha un errore di trasmissione proprio sul primo byte che indica la lunghezza del frame, il ricevitore non sarà in grado di recuperare l'informazione contenuta in esso. Supponiamo ad esempio, di avere un frame di 100 byte. Il primo byte dovrebbe essere pari dunque, a 100. Se, per errore, tale byte si trasforma in 98, per il ricevitore gli ultimi due byte del frame non faranno parte di esso ma del frame successivo. L'errore si propaga in modo disastroso. Infatti il 99esimo byte del frame verrebbe interpretato come il byte che contiene la lunghezza del frame successivo per cui il ricevitore non riuscirà a delimitare il secondo frame e così via.

Character stuffing

Un secondo tipo di formattazione del frame prevede che esso inizi con la sequenza di due caratteri ascii:

DLE (DATA LINK ESCAPE)

STX (START OF TEXT)

E termini con un'altra coppia di caratteri

DLE (DATA LINK ESCAPE)

ETX (END OF TEXT)

Esempio

I dati da trasferire sono i seguenti

55	A6	02	F7	25	0B	15

Tenendo presente che il codice di DLE è 10, quello di STX è 02 e quello di ETX è 03 il frame diventa

10	**02**	55	A6	02	F7	25	0B	15	**10**	**03**

Supponiamo ora che i dati da trasferire diventano i seguenti

55	A6	02	F7	10	03	15

Il frame diventa

10	**02**	55	A6	02	F7	**10**	**03**	15	**10**	**03**

Si nota come ci sono due byte di dati che coincidono con i codici di controllo finali. Il ricevitore potrebbe interpretarli, allora, come fine del frame. Per evitare quest'equivoco si usa la tecnica di character stuffing o riempimento di caratteri. In sostanza, ogni volta che nei dati appare un byte pari a 10 cioè DLE, viene aggiunto un altro byte pari a 10. nel nostro caso avremmo

10	**02**	55	A6	02	F7	**10**	**10**	**03**	15	**10**	**03**

Vedendo due byte pari a 10 che si seguono il ricevitore capisce che non si trova di fronte a caratteri di controllo ma a dati ed elimina il byte in più. Il solerte studente potrebbe chiedersi: cosa succede se nei dati da trasmetter vi sono già due byte pari a 10?

10	**02**	55	A6	02	F7	**10**	**10**	**03**	15	**10**	**03**

Nessun problema! Dopo ogni byte del dato pari a 10 il trasmettitore inserisce un nuovo byte pari a 10

10	**02**	55	A6	02	F7	**10**	**10**	**10**	**10**	**03**	15	**10**	**03**

In ogni caso, poiché il ricevitore sa che il trasmettitore inserirà sempre un byte pari a 10 ogni volta che incontra nei dati un byte pari a 10, interpreta le coppie 10, 10 come dati e non caratteri di controllo ed elimina il secondo byte della coppia, ricostruendo il frame originale.

Naturalmente, se vi è un errore di trasmissione che modifica un byte di controllo, il ricevitore non riesce a delimitare un frame. Però quando arriva una nuova coppia DLE STX esso capisce che inizia un nuovo frame e può richiedere il rinvio del frame precedente. Un altro difetto consiste nel fatto che questo tipo di formattazione dei frame si base sul codice ascii che prevede che i caratteri siano sempre espressi su 8 bit. Questo è un limite poiché impedisce di utilizzare tecniche di compattazione dei dati basate sul fatto che i vari caratteri possano essere espressi anche con un numero di bit variabili.

Bit stuffing

Si usa allora una formattazione orientata al bit in cui il campo dati non è rappresentato da una sequenza di byte ma una stringa di bit di lunghezza variabile. Il frame è delimitato da una stringa di bit pari a 01111110 all'inizio e alla fine

Se vogliamo inviare, ad esempio, la seguente stringa 0001110011, essa diventa **01111110** 0001110011 **01111110.** Cosa succede se, all'interno della stringa dei dati da trasmettere compare proprio una stringa uguale alla stringa delimitatrice? Ad esempio 00*01111110*011? Si avrebbe il seguente frame

0111111000*01111110*011**01111110**

Nell’esempio il ricevitore interpreterebbe la sequenza in corsivo come delimitatore del frame. Riterrebbe di aver ricevuto solo il frame costituito dalla sequenza 00. Per evitare che il ricevitore possa interpretare la stringa di bit dato come il delimitatore della fine del frame, si usa la tecnica del bit stuffing. In sostanza ogni volta che vi sono 5 bit 1 in sequenza, il trasmettitore inserisce uno zero

01111110*00111110**10***011**01111110.**

Nota bene: il trasmettitore inserisce sempre un bit pari a 0 dopo cinque bit pari ad 1. Anche se non ce n’è bisogno.

Esempio

01111110*00111110011***01111110**

diventa

01111110*00111110**00***011**01111110**

Nell’esempio non c’era possibilità di interpretazione scorretta ma il bit 0 aggiuntivo viene inserito ugualmente per semplificare il metodo. In ogni caso il ricevitore sa che dopo cinque bit ad 1, il sesto bit va eliminato perché non fa parte dei dati. In ogni caso, con il bit stuffing non potrà mai succedere che il campo dati possa contenere una stringa di bit uguale alla stringa delimitatrice.

Protocolli di comunicazione

Nella comunicazione fra trasmettitore e ricevitore si devono risolvere i problemi di sincronizzazione che consistono nel garantire la corrette trasmissione dei frame anche quando trasmettitore e ricevitore lavorano a velocità diverse ed inoltre garantire un

controllo che informi il trasmettitore che il ricevitore ha ricevuto un frame in modo corretto ed, in caso contrario, gli rinvii il frame.

Protocolli di tipo simplex

I dati possono andare soltanto in una direzione. All'attivazione del collegamento viene definito quale nodo funzione funziona da trasmettitore e quale da ricevitore. Il ricevitore, da questo momento in poi, può soltanto ricevere dati e spedire al massimo caratteri di conferma.

Simplex non limitato

E' il protocollo più semplice poiché presuppone che trasmettitore e ricevitore funzionino alla stessa velocità e che il canale sia a prova di errore. In sostanza non si pone il problema di sincronizzare ad esempio, un trasmettitore più veloce ed un ricevitore più lento né si pone il problema del controllo sulla correttezza della trasmissione. In sostanza questo protocollo, in trasmissione, si occupa soltanto di prelevare il pacchetto dallo strato di trasporto, realizzare il frame ed inviarlo, per la spedizione, allo strato fisico mentre in ricezione, preleva il frame dallo strato fisico, ne estrae il pacchetto e lo invia allo strato di rete.

Simplex stop and wait

Nel caso che il trasmettitore ed il ricevitore lavorino a velocità diverse il protocollo precedente non funziona. Il trasmettitore non può inviare un nuovo frame se il ricevitore non è riuscito ancora ad inviare al suo strato di rete il pacchetto contenuto nel frame precedente. La soluzione più semplice è che il trasmettitore, dopo aver inviato un frame esegue un certo numero calcolato a priori di cicli di attesa per dare il

tempo al ricevitore di completare il trattamento del frame che gli ha inviato precedentemente. Il problema è che risulta difficile calcolare a priori la lentezza del ricevitore e poi la rete risulterebbe rallentata complessivamente dalla velocità del nodo più lento. Una soluzione migliore è il protocollo stop and wait in cui il trasmettitore, dopo aver inviato un frame, attende un frame di riscontro dal ricevitore che gli da, in questo modo, via libera per l'invio di un nuovo frame.

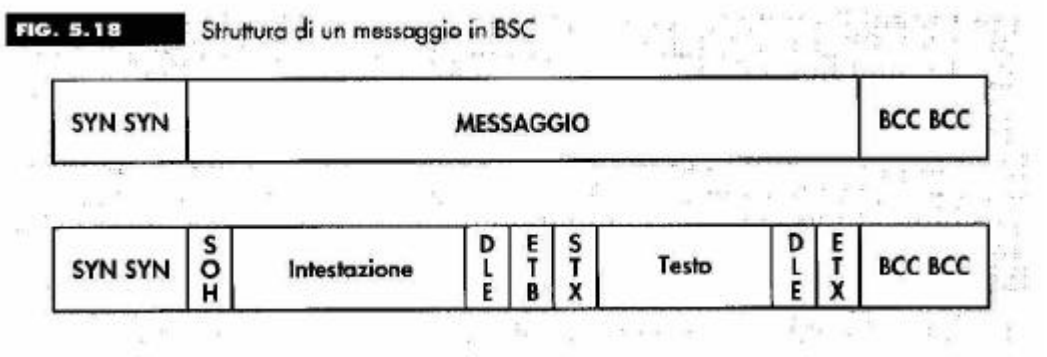
Protocolli a controllo e ritrasmissione

I protocolli visti non prevedono la gestione di eventuali errori nella trasmissione. Una prima soluzione è quella di modificare il protocollo stop and wait ed introdurre nel frame dei byte di checksum. Il ricevitore, se riscontra, tramite il confronto dei checksum, l'assenza di errori, invia al trasmettitore un messaggio di accettazione. Se vi è stato un errore il messaggio di acknowledged non viene inviato. In tal caso il trasmettitore invia di nuovo il frame. Il problema può sorgere dal fatto che gli errori di trasmissione riguardino proprio il messaggio ACK del ricevitore che potrebbe andare perso. In tal caso il trasmettitore invia una seconda volta il frame pensando che il precedente sia giunto danneggiato. Il ricevitore lo interpreterebbe come un nuovo frame mentre, in realtà, si tratta del precedente frame che è stato ritrasmesso. Nei protocolli PAR (positive Acknowledgment with retransmission) il problema viene risolto poiché alla struttura del frame viene aggiunto un campo che segnala se il frame inviato è quello originale o una copia inviata dal trasmettitore perché ha ritenuto che l'originale non sia stato ricevuto correttamente.

Protocollo BSC (Binary Synchronous Communication)

Questo protocollo ha diverse varianti. Le BSC1 e BSC2 consentono lo comunicazione punto-punto, la BSC3 consente la comunicazione multipunto

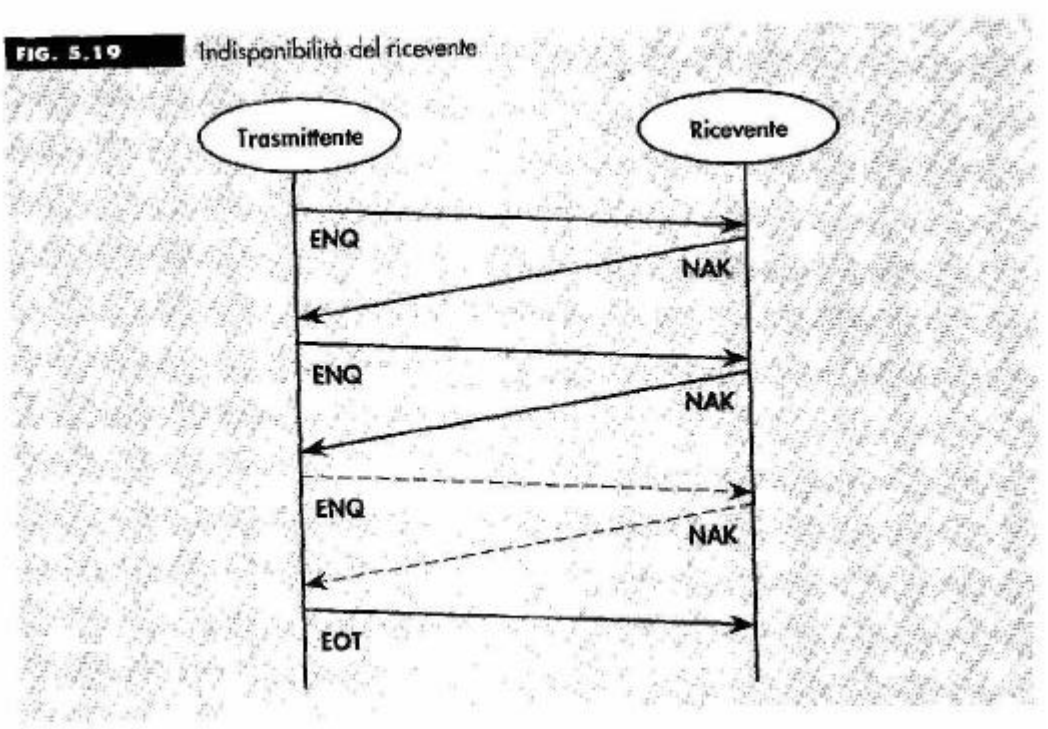
La struttura del frame del protocollo BSC è la seguente



i caratteri SYN e BCC delimitano il frame. Nel frame abbiamo un campo intestazione che contiene gli indirizzi della stazione o dei dispositivi cui è diretto il messaggio. L'intestazione è delimitata dai caratteri SOH (Start of Header) e DLE ETB (Data Link Escape- End of Block). L'informazione vera e propria è racchiusa nel campo testo ed è delimitata dai caratteri STX (Start Of Txt) e DLE ETX (Data Link Escape End Of Text). Vediamo ora le regole che segue questo protocollo.

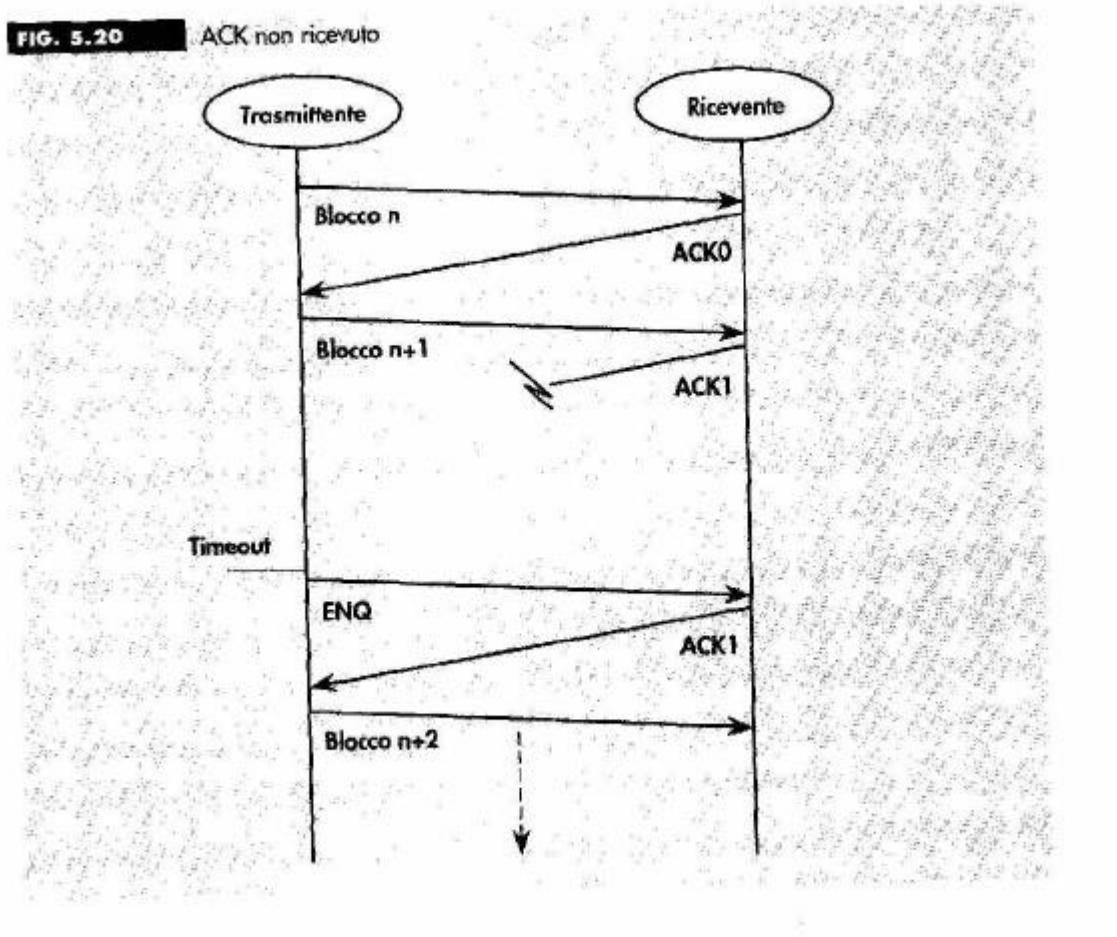
Il trasmettitore cerca di stabilire una comunicazione con il ricevitore inviandogli un messaggio di controllo ENQ (Enquiry).

Se il ricevitore non è disponibile, risponde con un messaggio Not Acknowledge NAK. Il trasmettitore riprova più volte ad inviare un ENQ. Dopo un certo lasso di tempo di timeout, se non è riuscito a collegarsi con il ricevitore, il trasmettitore interrompe il collegamento.

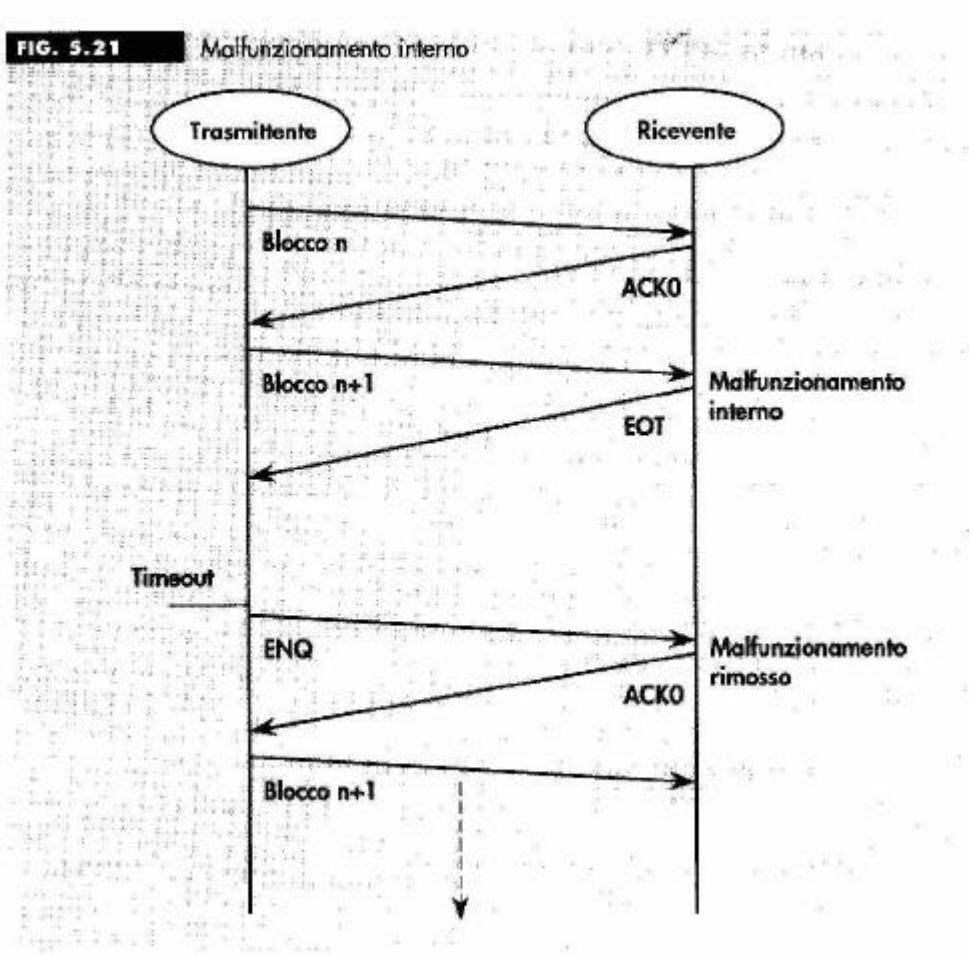


Se il ricevitore, invece, è disponibile al dialogo, risponde all'Enquiry mediante un messaggio di Acknowledge ACK. Nel BSC sono previsti due messaggi di riscontro ACK0 e ACK1. Questi due messaggi vengono inviati alternativamente e questo meccanismo consente di riscontrare la presenza di problemi di trasmissione. Supponiamo ad esempio che il trasmettitore invii il blocco numero N e che il ricevitore abbia risposto con il messaggio ACK0. Ora viene trasmesso il blocco N+1 a cui il ricevitore dovrebbe rispondere con il messaggio ACK1. Per un errore di trasmissione tale messaggio non viene ricevuto dal trasmettitore. Il trasmettitore attende il messaggio di riscontro per un tempo di timeout e poi invia un messaggio di ENQ. Il ricevitore reagisce inviando di nuovo l'ultimo messaggio di riscontro che aveva inviato, nel nostro caso ACK1. Vedendo che è stato spedito il messaggio di riscontro ACK1 il trasmettitore capisce che il blocco N+1 era stato ricevuto correttamente da parte del ricevitore, e che l'errore di trasmissione si riferisce

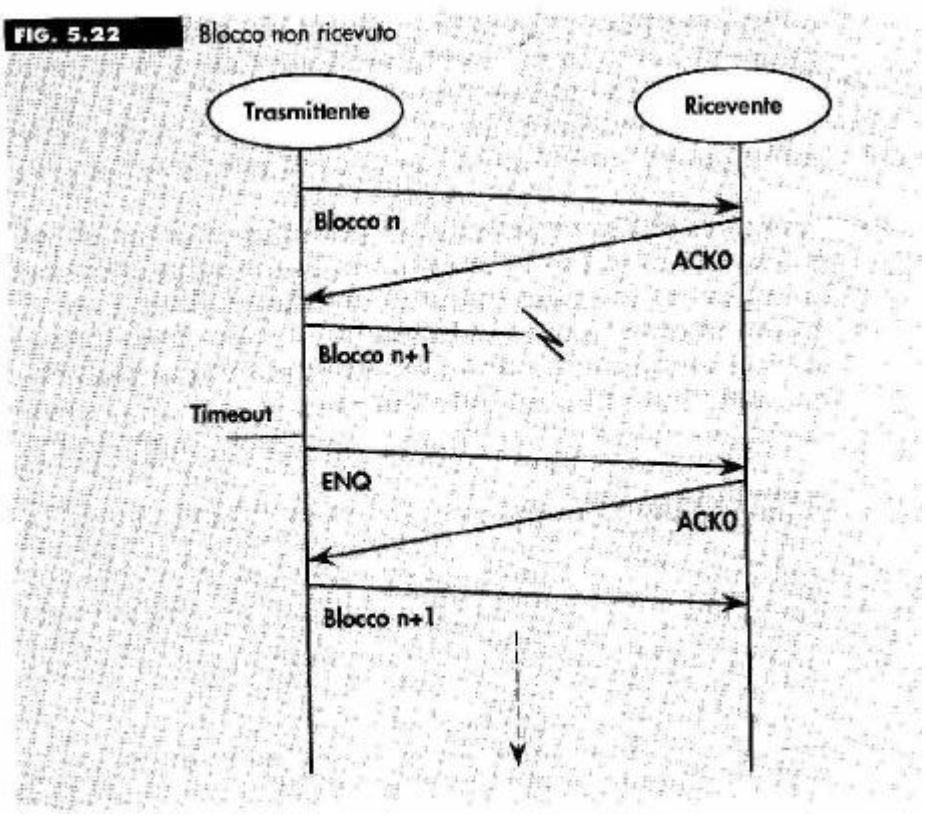
soltanto al pacchetto di riscontro, per cui provvede a trasmettere normalmente il blocco N+2



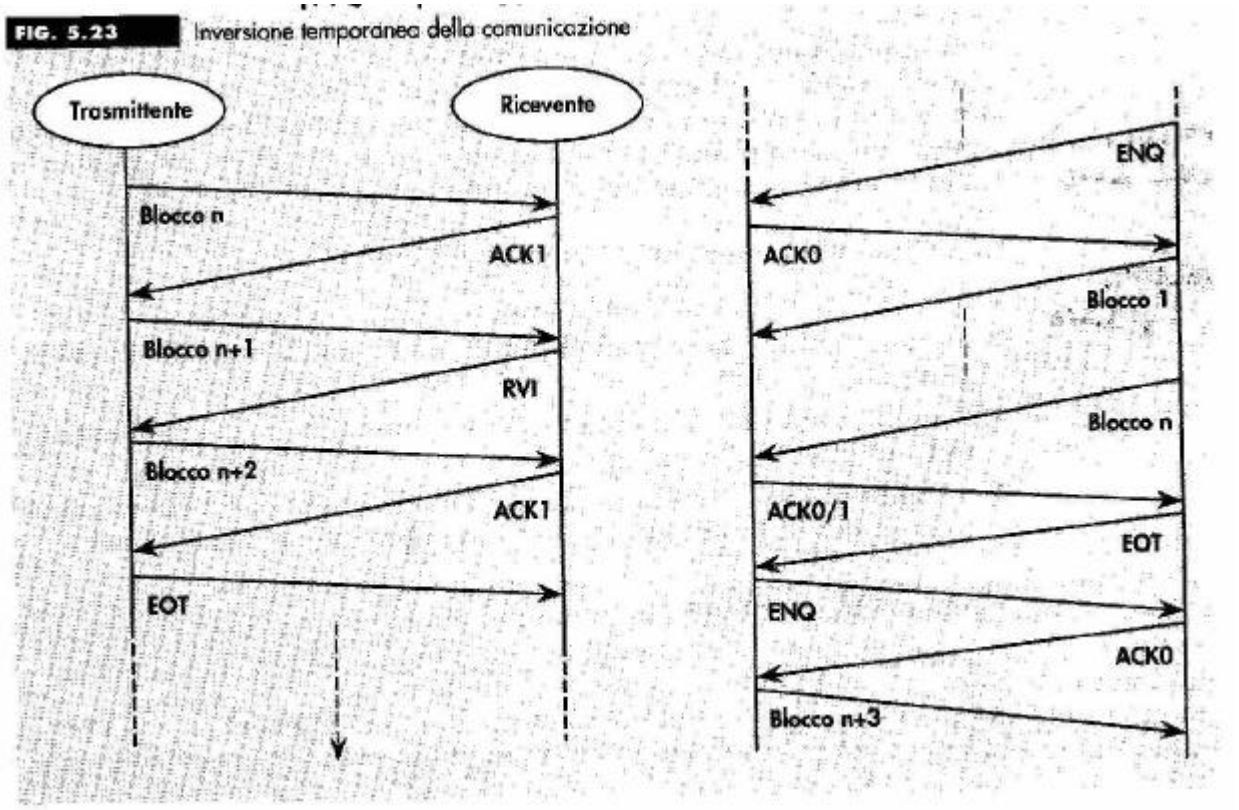
Supponiamo ora che, a causa di un malfunzionamento, il ricevitore non riceva il blocco N+1 e tenti di scollegarsi per un malfunzionamento interno, inviando il messaggio EOT (End of transmission). In tal caso, passato un tempo di timeout, il trasmettitore tenta di nuovo di ricollegarsi al ricevitore con un messaggio di ENQ. Il ricevitore, se di nuovo disponibile, invia l'ultimo messaggio di riscontro che aveva inviato, nell'esempio ACK0. Da ciò il trasmettitore capisce che il blocco N+1 non è stato ricevuto e lo ritrasmette



Un funzionamento analogo si ha nel caso che il blocco N+1 venga perso per un disturbo sulla linea

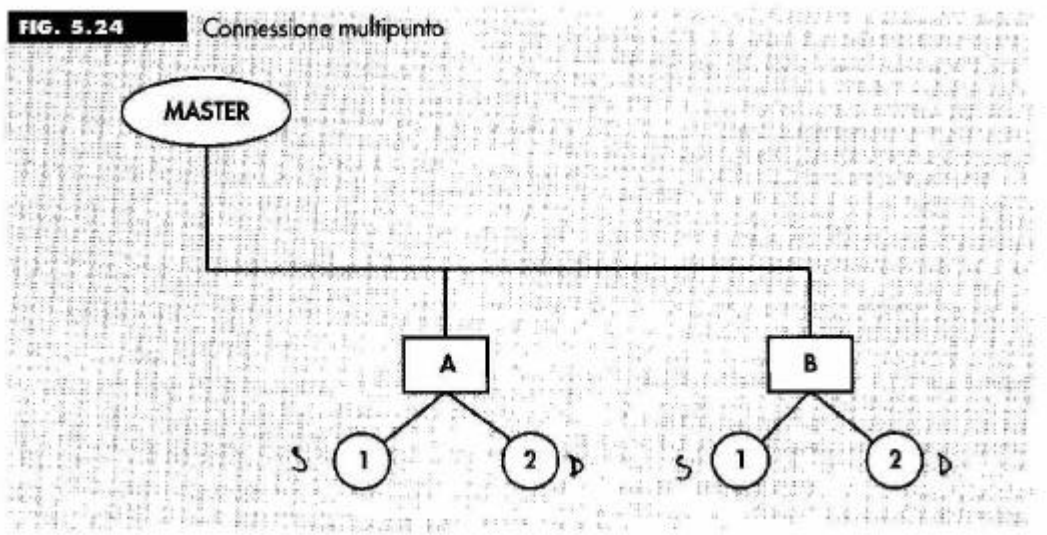


il diagramma seguente mostra, invece, come viene gestita l'inversione del senso di comunicazione. Il ricevitore, per diventare il nuovo trasmettitore invia il messaggio RVI (reverse Interrupt). Il trasmettitore termina il suo invio di dati mediante il messaggio di EOT. Il ricevitore diventa il nuovo trasmettitore ed invia alla prima stazione un segnale di Enquiry. Da questo momento in poi la procedura è identica.



BSC3

Il protocollo BSC3 consente la gestione del collegamento multipunto

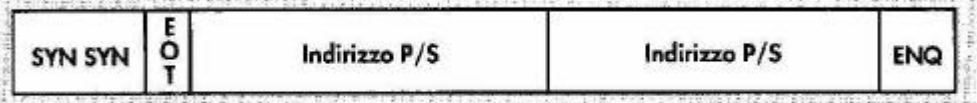


in questo caso vi è un master che controlla più slave.

Quando il master vuole inviare i dati ad uno degli slave effettua un’operazione di selecting. Se invece interroga le stazioni subordinate per vedere quale di esse voglia

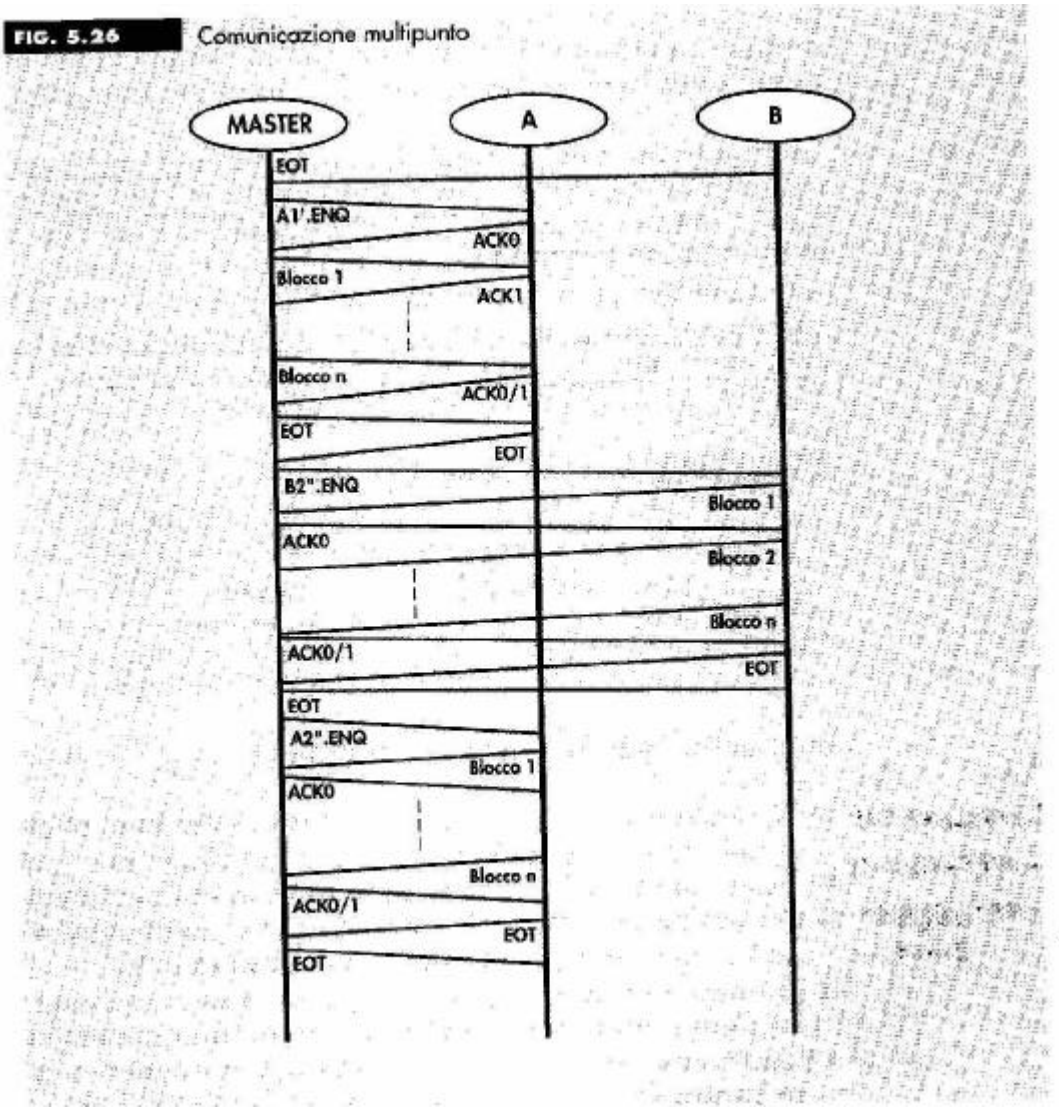
inviargli dei dati effettua un polling. Nella figura supponiamo che il master sia collegato a due slave A e B. per ognuno dei due slave vi è un dispositivo 1 che è una stampante ed un dispositivo 2 che è un disco rigido. Il frame del BSC3 è il seguente

FIG. 5.25 Struttura di un messaggio in BSC3



nel caso di un messaggio di comando agli slave. Il carattere EOT indica che si tratta di un comando. Il primo indirizzo è quello della stazione slave, il secondo indirizzo è quello del dispositivo collegato allo slave. Studiamo allora il diagramma seguente come esempio

FIG. 5.26 Comunicazione multipunto



il master vuole scrivere sulla stampante collegata allo slave A. invia allora un Enquiry all'indirizzo A1 corrispondente alla stampante 1 dello slave A. segue un normale processo di invio blocchi e di messaggi di riscontro ACK. Termina il collegamento mediante lo scambio di messaggi EOT. Poi il master tenta di leggere informazioni dal disco rigido collegato allo slave B. Invia allora un Enquiry all'indirizzo B2 (dispositivo 2 dello slave B). Questo inizia ad inviare blocchi di dati al aster che risponde con messaggi di riscontro e così via.