Calcolatori Elettronici
Interfacce

Ing. Gestionale e delle Telecomunicazioni

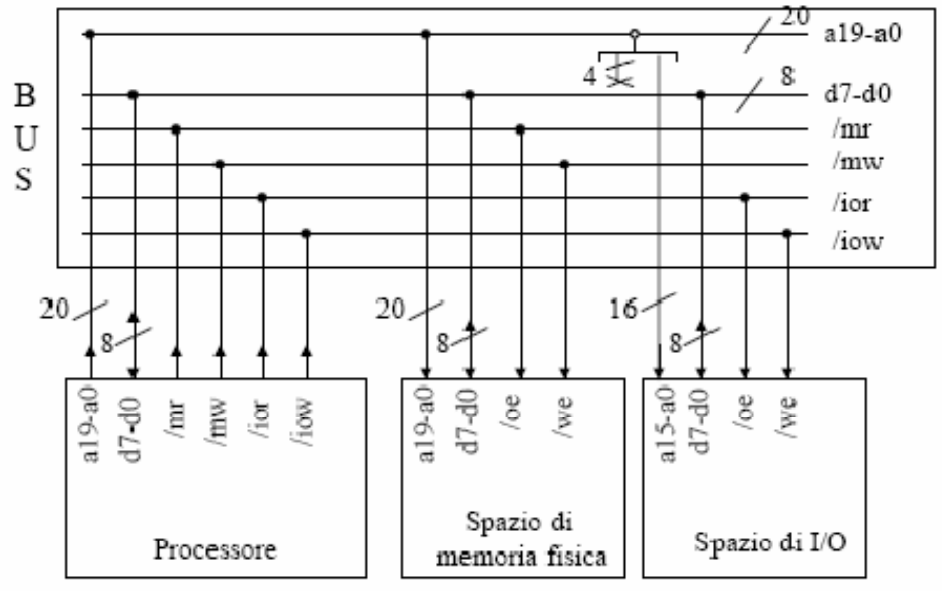
A.A. 2008/09

Gabriele Cecchetti

Sommario

- Organizzazione dello spazio di I/O
- Interfacce parallele di ingresso/uscita
  - senza *handshake*
  - con *handshake*
- Interfacce seriali

Schema di un calcolatore elementare di riferimento

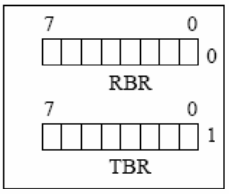


Organizzazione dello spazio di I/O

- Lo spazio di I/O è implementato mediante circuiti detti interfacce, che sono connessi sia al bus che ai trasduttori.
- Abbiamo utilizzato 16 delle 20 variabili per gli indirizzi, pertanto lo spazio di I/O e' dotato di 64K locazioni

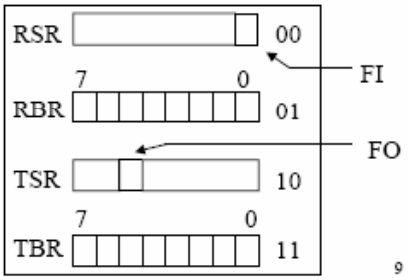
Schema funzionale di una semplice interfaccia

- L'interfaccia contiene **due registri da 8 bit** mediante i quali implementa due porte dello spazio di I/O.
- L'indirizzo di tali porte dipende da una maschera che affianca l'interfaccia.
- **Receiver Buffer Register** (**RBR**): contiene l'ultimo byte che l'interfaccia ha prelevato dal trasduttore esterno. RBR è accessibile in lettura al processore.
- ***Transmitter Buffer Register*** (**TBR**): il byte che contiene è reso disponibile al trasduttore esterno. TBR è accessibile in scrittura al processore.



Schema funzionale di una interfaccia con registri di stato (1/2)

- Il precedente tipo di interfaccia non consente alcuna sincronizzazione tra il processore ed il trasduttore esterno:
  - quando il processore preleva il contenuto di RBR non può sapere se si tratta di un nuovo byte
  - quando immette un nuovo dato in TBR non può sapere se il dato precedente e' stato "consumato" dal trasduttore



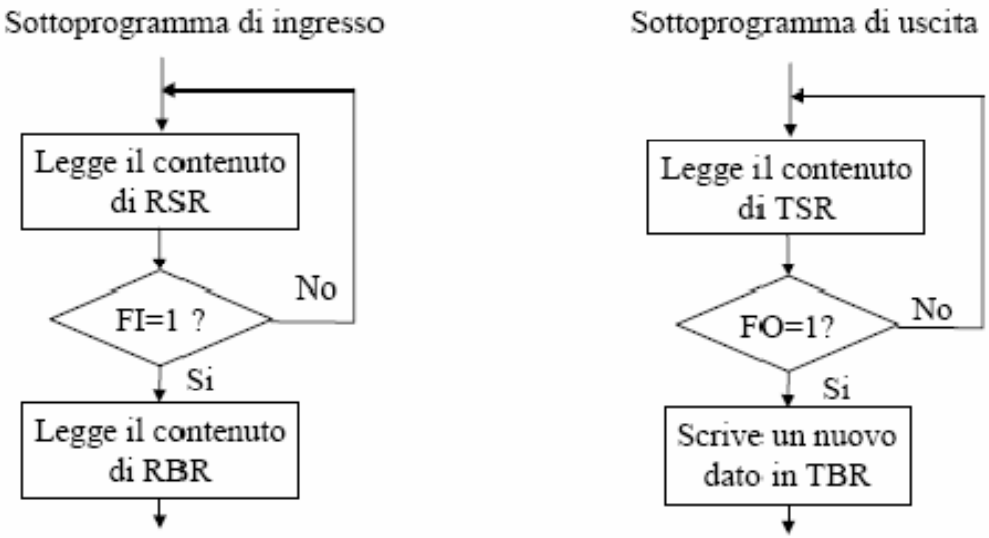
FI: flag di buffer di ingresso pieno

FO: flag di buffer di uscita vuoto

Schema funzionale di una interfaccia con registri di stato (2/2)

- **`FI=1`** indica che **l'interfaccia ha ricevuto un nuovo byte** dal trasduttore e che tale byte è disponibile al processore nel registro RBR.
  - *Quando il processore preleva il dato da RBR, mediante una operazione di lettura, l'interfaccia mette automaticamente a `0` il valore di `FI`* ed è pronta a ricevere un nuovo byte dal trasduttore.
- **`FO=1`** indica che **il byte attualmente contenuto in TBR è stato prelevato dal trasduttore** e che l'interfaccia è disponibile ad accettare un nuovo dato.
  - *Quando il processore fornisce un nuovo dato all'interfaccia, mediante una operazione di scrittura, l'interfaccia mette automaticamente a `0` il valore di `FO`*.

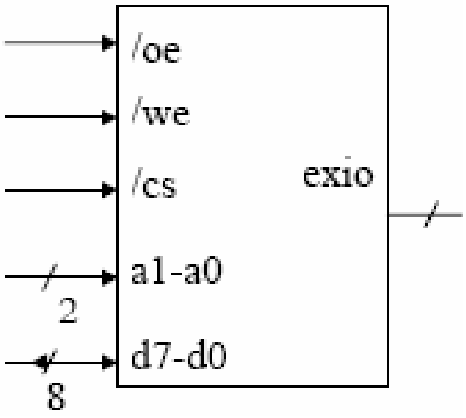
I/O a controllo di programma



- Il processore spreca tempo per sincronizzarsi con il trasduttore esterno.

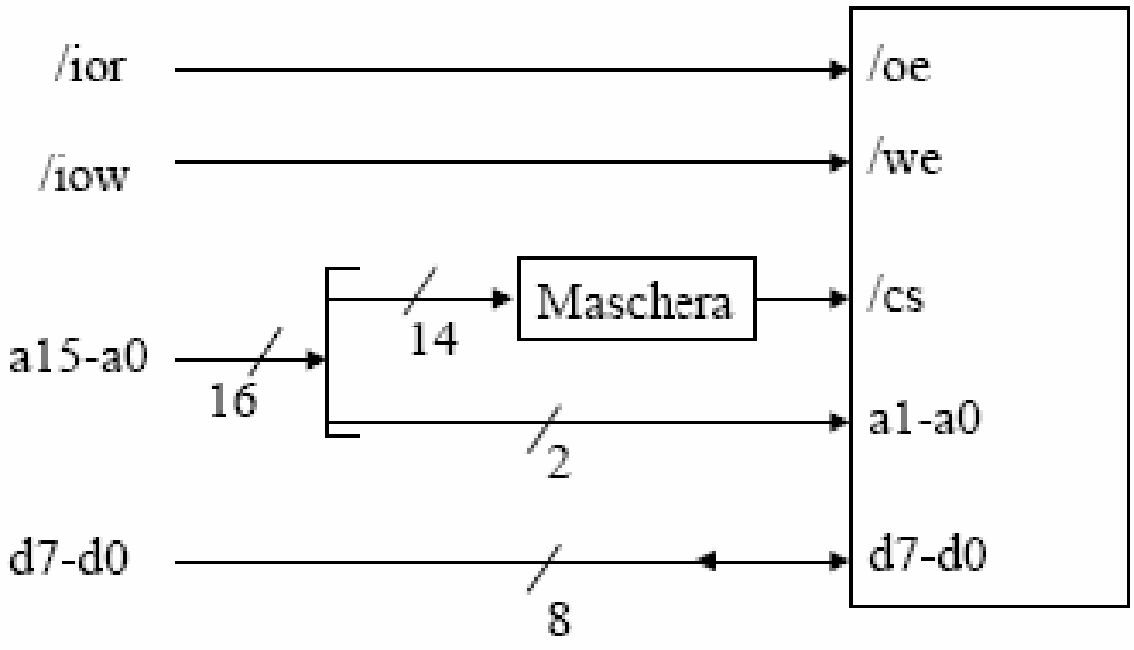
Interfaccia: variabili di ingresso e uscita

/cs	/we	/oe	
1	-	-	: nessuna azione
0	1	1	: nessuna azione
0	1	0	: ciclo di lettura
0	0	1	: ciclo di scrittura
0	0	0	: non definito



- ***a1-a0*** : indirizzo interno del registro coinvolto nel ciclo di lettura o di scrittura
- ***d7-d0*** : Variabili di uscita durante un ciclo di lettura, variabili di ingresso durante un ciclo di scrittura. Altrimenti in alta impedenza.
- ***exio*** : Utilizzate per lo scambio di informazioni con il trasduttore. Fortemente dipendenti dal tipo di interfaccia.

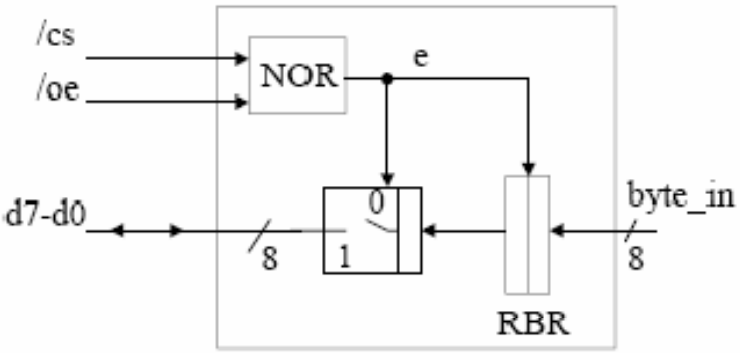
Interfaccia: modulo di espansione



Interfacce parallele

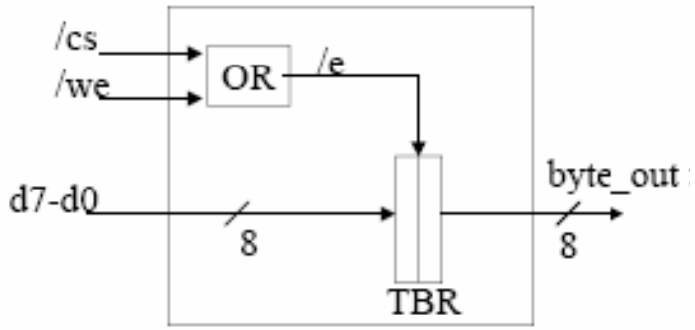
- Le interfacce parallele gestiscono trasduttori in grado di trasferire in parallelo più bit.
- Possono essere classificate in:
  - *interfacce parallele senza handshake*: non consentono la sincronizzazione tra processore e trasduttore;
  - *interfacce parallele con handshake*: consentono tale sincronizzazione.

Interfaccia parallela di ingresso senza *handshake*



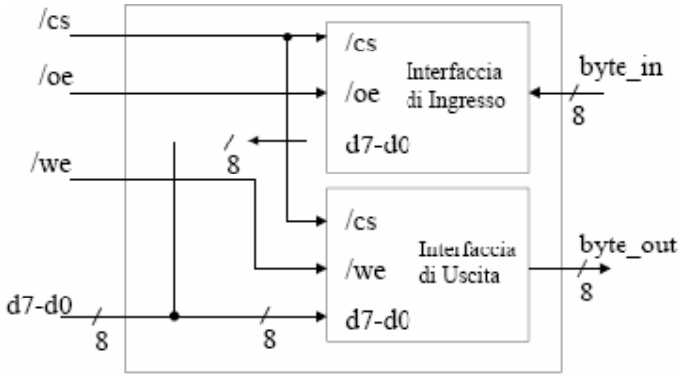
- La maschera che genera **`/cs`** riceve in ingresso tutte e 16 le variabili degli indirizzi e la configurazione per la quale genera **`0`** corrisponde all'indirizzo nello spazio di I/O del registro RBR.
- *Quando l'interfaccia è selezionata* (**`/cs=0`**) *e inizia un ciclo di lettura* (**`/oe` va a zero**), allora '**`e`**' passa da **`0`** a **`1`** ed il registro RBR memorizza il valore delle variabili **`byte_in`**. Inoltre le porte 3-state passano in conduzione.

Interfaccia parallela di uscita senza *handshake*



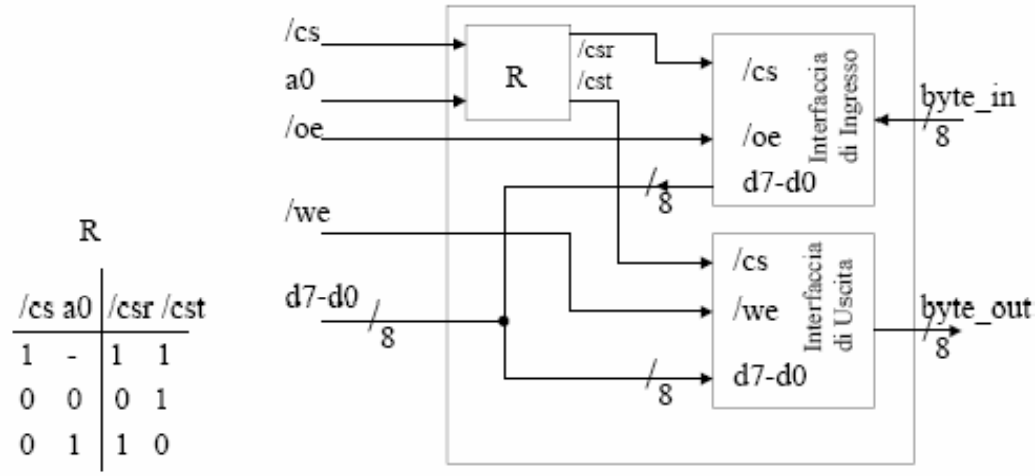
- La maschera che genera **/cs** riceve in ingresso tutte e 16 le variabili degli indirizzi e la configurazione per la quale genera 0 corrisponde direttamente all'indirizzo nello spazio di I/O del registro TBR
- Quando l'interfaccia è selezionata (**`/cs=0`**) e
  - inizia un ciclo di scrittura (**`/we` va a 0**), anche **`/e` passa da 1 a 0**;
  - quando finisce il ciclo di scrittura **`/we` torna a 1** ed **`/e` passa a 1**, quindi il registro TBR memorizza il byte presentato dal processore.

Interfaccia parallela di ingresso-uscita senza *handshake*



- I due registri **RBR** e **TBR** implementano la stessa porta dello spazio di I/O:
  - se il ciclo e' di lettura viene coinvolto il registro **RBR**,
  - se il ciclo e' di scrittura viene coinvolto il registro **TBR**.
- Da un punto di vista funzionale è come se l'interfaccia avesse un unico registro **RTBR**

Interfaccia parallela di ingresso-uscita senza *handshake*



R

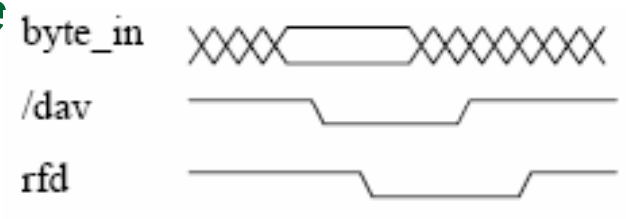
/cs	a0	/csr	/cst
1	-	1	1
0	0	0	1
0	1	1	0

- Interfaccia parallela di ingresso/uscita che mantiene la distinzione tra i registri RBR e TBR. Il registro viene selezionato mediante **`a0`**.

Interfacce parallele con handshake

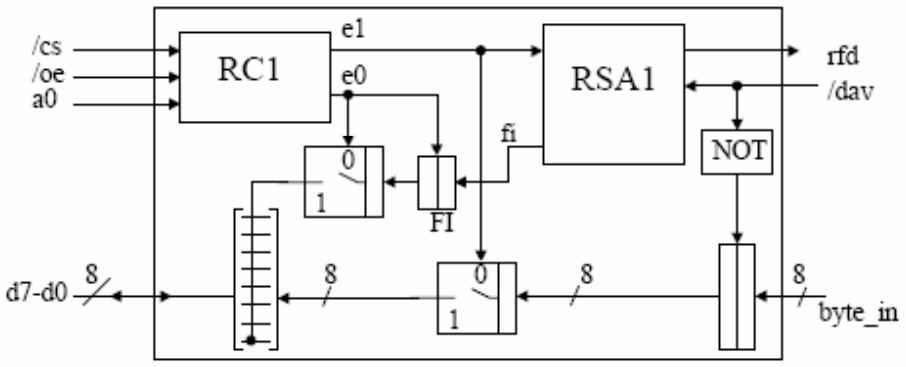
- Le interfacce parallele con *handshake* sono dotate di due variabili, ***`rfd`*** e ***`/dav`***, che consentono di colloquiare con il trasduttore.
- Le variabili ***`/dav`*** e ***`rfd`*** sono rispettivamente di ingresso e di uscita per l'interfaccia di ingresso, viceversa per l'interfaccia di uscita.
- L'*handshake* è gestito da una rete sequenziale asincrona che gestisce anche le variabili **`FI`** ed **`FO`**.

Interfaccia parallela di ingresso con *handshake*



- Situazione iniziale:
  - ***rfd* = 1** l'interfaccia è disponibile a prelevare un dato;
  - ***/dav* = 1** nessun dato utile è stato presentato dal trasduttore all'interfaccia.
- Il trasduttore presenta un byte utile come stato delle variabili ***byte_in*** e pone ***/dav* a 0**.
- L'interfaccia preleva il byte utile e lo memorizza nel registro RBR, quindi pone ***rfd* a 0**.
- Il trasduttore riporta ***/dav* a 1** ed attende che l'interfaccia riporti ***rfd* a 1** ad indicare la disponibilità ad accettare un nuovo dato.

Interfaccia parallela di ingresso con *handshake*



/cs	/oe	a0	e1	e0
0	0	0	0	1
0	0	1	1	0
others			0	0

RC1

Interfaccia parallela di ingresso con *handshake* (1/2)

e1 /dav	00	01	11	10	rfd	fi
S0	S1	(S0)	-	-	1	0
S1	(S1)	(S1)	S2	S2	0	1
S2	(S2)	S0	(S2)	(S2)	0	0

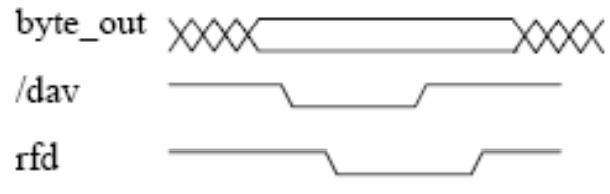
1. Stato iniziale S0 per **`e1 a 0`** e **`/dav a 1`** (*non è in corso un ciclo di lettura del registro **RBR** e nessun dato è stato ancora presentato dal trasduttore*).
   - La rete tiene **`fi a 0`** e **`rfd a 1`** (*nessun byte utile è disponibile per il processore* e *l'interfaccia è disponibile a ricevere un byte*).
2. Il trasduttore, dopo aver presentato un byte, pone `/dev` a `0`.
   - La rete passa nello stato S1 in cui **`fi va a 1`** (*un nuovo dato è disponibile per il processore*), e **`rfd a 0`** (*l'interfaccia non può ricevere altri byte*).

Interfaccia parallela di ingresso con *handshake* (2/2)

e1 /dav	00	01	11	10	rfd	fi
S0	S1	(S0)	-	-	1	0
S1	(S1)	(S1)	S2	S2	0	1
S2	(S2)	S0	(S2)	(S2)	0	0

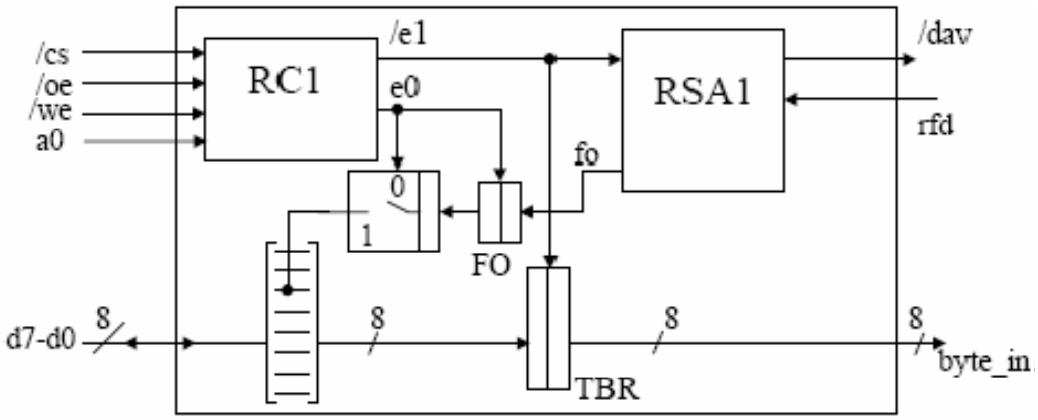
3. Quando il processore compie un ciclo di lettura del registro RBR, **`e1 va a 1`** e la rete si porta nello stato S2 in cui mette **`fi a 0`**.
4. Quando il ciclo di lettura termina (**`e1 torna a 0`**) e il trasduttore ha riportato **`/dav a 1`**, la rete torna nello stato iniziale S0 (in cui completa l'*handshake* riportando **`rfd a 1`**).

Interfaccia parallela di uscita con *handshake:* temporizzazione



- Situazione iniziale:
  - ***rfd* = 1** trasduttore è disponibile a prelevare un dato;
  - ***/dav* = 1** nessun dato utile è contenuto nel registro TBR.
- L'interfaccia presenta un byte utile come stato delle variabili ***byte_out*** e pone ***/dav* a 0.**
- Il trasduttore preleva il byte utile, quindi pone ***rfd* a 0.**
- L'interfaccia riporta ***/dav* a 1** ed attende che il trasduttore riporti ***rfd* a 1** ad indicare la sua disponibilità ad accettare un nuovo dato.

Interfaccia parallela di uscita con *handshake:* schema funzionale



/cs	/we	/oe	a0	/e1	e0	
0	1	0	0	1	1	RC1
0	0	1	1	0	0	
others				1	0	

Interfaccia parallela di uscita con *handshake:* descrizione RSA (1/4)

- Si parte dallo stato stabile S0 per **`/e1 a 1`** e **`rfd a 1`** (*non è in corso un ciclo di scrittura del registro TBR* e *il trasduttore è disponibile a ricevere un byte*).
  - La rete tiene **`fo a 1`** e **`/dav a 1`** (*il processore può scrivere un nuovo dato in TBR* e *nessun dato utile è presente come stato delle variabili* ***`byte_out`***).

/e1 rfd	00	01	11	10	/dav	fo
S0	-	S1	(S0)	-	1	1
S1	-	(S1)	S2	-	1	0
S2	-	-	(S2)	S3	0	0
S3	-	-	S0	(S3)	1	0

Interfaccia parallela di uscita con *handshake:* descrizione RSA (2/4)

- Quando il processore compie un ciclo di scrittura in TBR, la variabile **`/e1` va prima a `0`** (la rete si porta in S1) **e poi torna ad `1`** (la rete si porta in S2).
  - Il dato viene memorizzato in TBR.
- In S1 e S2 la rete tiene **`fo a 0`** *per indicare al processore che non è possibile scrivere un altro dato*. In S2, **`/dav` viene portato a `0`** *per indicare al trasduttore la presenza di un dato valido*.

/e1 rfd	00	01	11	10	/dav	fo
S0	-	S1	(S0)	-	1	1
S1	-	(S1)	S2	-	1	0
S2	-	-	(S2)	S3	0	0
S3	-	-	S0	(S3)	1	0

Interfaccia parallela di uscita con *handshake:* descrizione RSA (3/4)

- Il trasduttore preleva il dato e **porta `rfd` a `0`**, facendo transire la rete nello stato S3 in cui **`/dav` viene riportato ad `1`**.

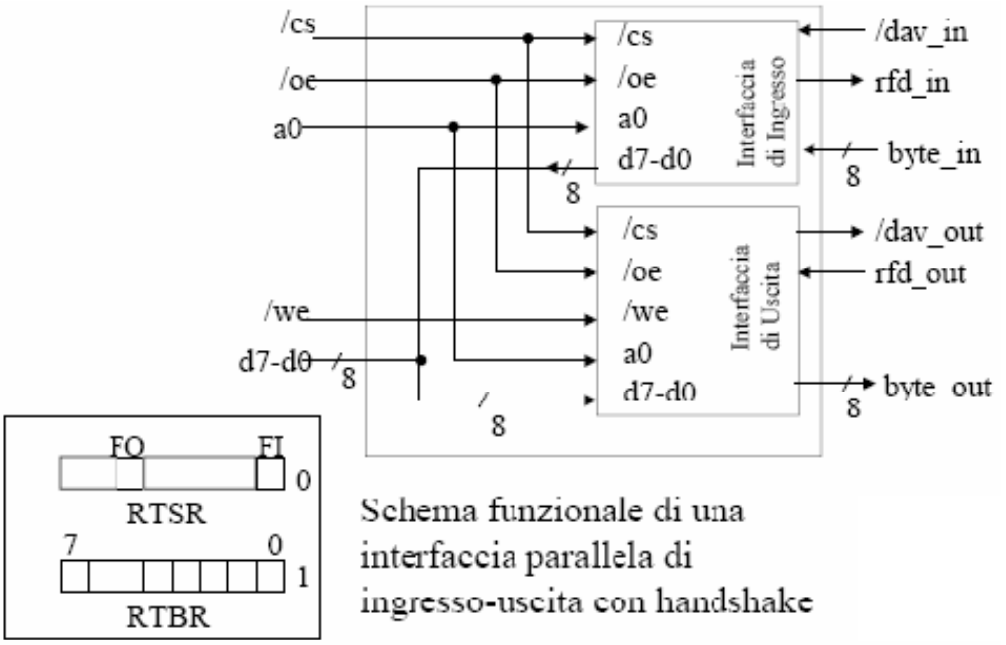
/e1 rfd	00	01	11	10	/dav	fo
S0	-	S1	(S0)	-	1	1
S1	-	(S1)	S2	-	1	0
S2	-	-	(S2)	S3	0	0
S3	-	-	S0	(S3)	1	0

Interfaccia parallela di uscita con *handshake:* descrizione RSA (4/4)

- Quando il trasduttore è disponibile a ricevere un altro dato riporta **`rfd` a `1`**, facendo tornare la rete nello stato iniziale S0 in cui **`fo` viene riportato ad `1`**.

/e1 rfd	00	01	11	10	/dav	fo
S0	-	S1	(S0)	-	1	1
S1	-	(S1)	S2	-	1	0
S2	-	-	(S2)	S3	0	0
S3	-	-	S0	(S3)	1	0

Interfaccia parallela di ingresso-uscita con *handshake:* schema funzionale



Schema funzionale di una interfaccia parallela di ingresso-uscita con handshake

Interfaccia seriale: generalità

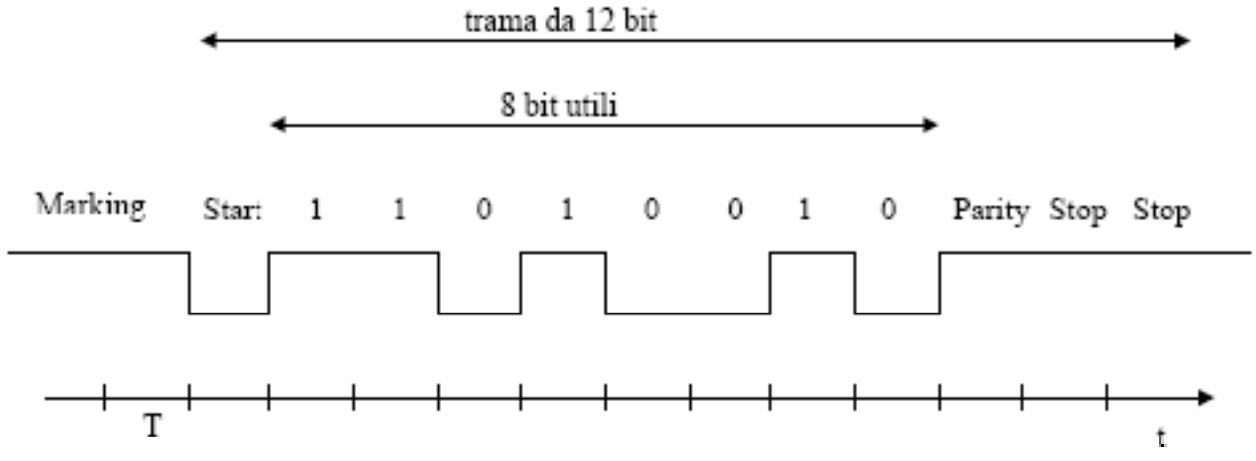
- **Comunicazione seriale asincrona**: un dispositivo trasmettitore ed un ricevitore sono in grado di scambiare dati mediante una sola linea di collegamento sulla quale viaggiano serialmente i singoli bit.
- I bit sono trasmessi e ricevuti in gruppi detti ***trame***.
- Durante la trasmissione di una trama i bit sono trasmessi con cadenza regolare (l'intervallo di un tempo T tra un bit e l'altro e' detto ***tempo di bit***).
- *L'intervallo di tempo che intercorre tra la fine di una trama e l'inizio della successiva non è soggetto a vincoli*.
- ***Bit-rate***: numero di bit inviati nell'unità di tempo durante la trasmissione di una trama (1/T).

Interfaccia seriale: trama

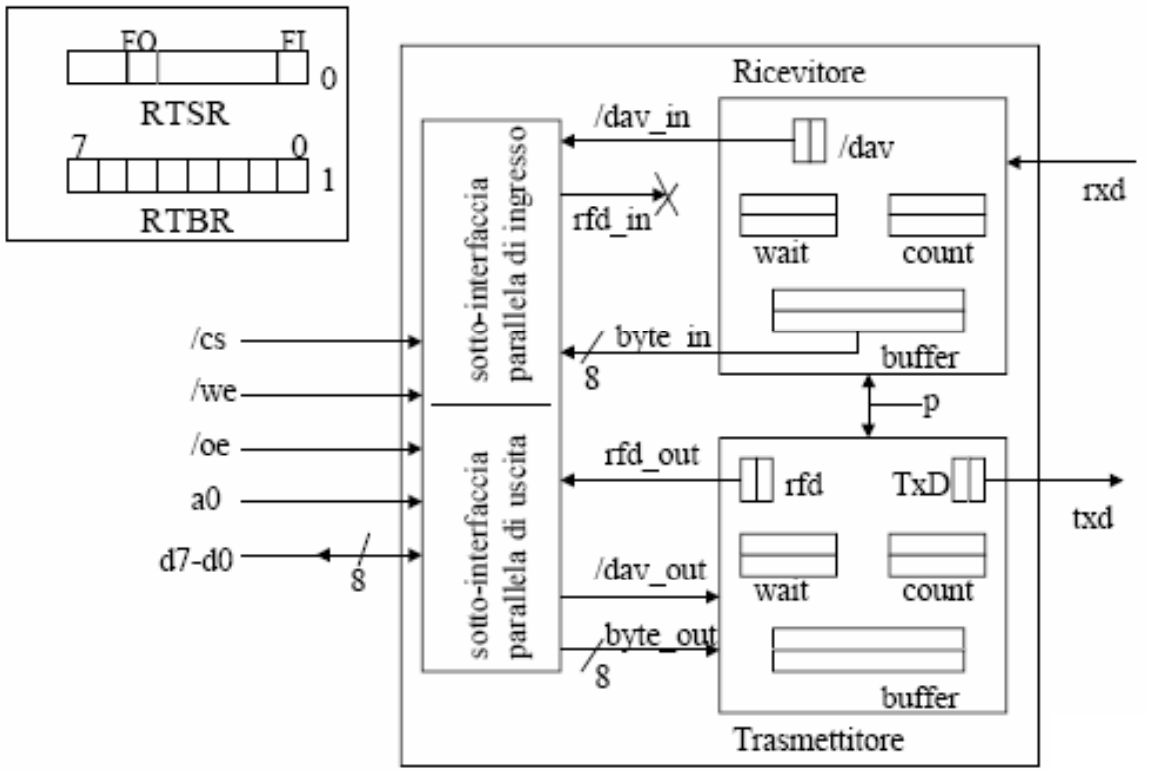
- Una trama è composta da un numero di bit che va da 7 a 12:
  - un *bit di start;*
  - da 5 a 8 bit utili (l'informazione vera e propria);
  - un eventuale *bit di parità;*
  - uno o due *bit di stop.*
- Tra una trama e l'altra la linea viene mantenuta nello *stato di marking.*
- Per trasmettere il bit di start si porta la linea nello *stato di spacing.*
- I bit di stop vengono trasmessi mantenendo la linea nello stato di *marking.*

Interfaccia seriale: esempio di trama

- I bit utili vengono trasmessi portando la linea nello stato di *marking* (1) o nello stato di *spacing* (0).



Interfaccia seriale: schema funzionale



Interfaccia seriale: il ricevitore

1. Situazione iniziale: **`/dav_in=1`** e **`count=8`** (trama con 8 bit utili).
2. Il ricevitore attende l'arrivo di un bit di start sulla variabile di ingresso **`rxd`**.
3. Attende un tempo pari a 1,5 T in modo tale da memorizzare il primo bit utile quando è arrivato da un tempo pari a T/2 (*centratura del bit*).
4. Preleva tutti bit utili ad intervalli di tempo pari a T (*il registro **`count`** è utilizzato per tener conto dei bit ricevuti*).
   Ogni volta che viene ricevuto un bit:
   1. il contenuto del registro buffer viene traslato a dx di una posizione;
   2. il bit viene immesso nel registro buffer come il suo bit più significativo.
5. Quando tutti i bit sono stati ricevuti, il ricevitore porta **`/dav_in a 0`** per notificare all'interfaccia di ingresso la presenza di un nuovo dato.
6. Attende un intervallo di tempo T in modo che arrivi il bit di stop e si riporta nello stato iniziale.

Interfaccia seriale: il trasmettitore

1. Il trasmettitore è inizialmente in una situazione di riposo con **`rfd_out=1`**, e attende che il valore della variabile di ingresso **`/dav_out`** vada a `0`.
2. A questo punto il trasmettitore mette **`rfd_out a 0`** e preleva il byte utile.
3. Costruisce la trama, aggiungendo ai bit utili il bit di start, un eventuale bit di parità e uno o più bit di stop. Deposita la trama nel registro buffer.
4. Trasmette la trama, un bit per volta, utilizzando il registro **`TxD`**.
5. Completa l'*handshake* con la sotto interfaccia di uscita .
6. Il registro **`count`** viene utilizzato per effettuare il conteggio dei bit trasmessi ed il registro **`wait`** per trasmettere i bit con la dovuta cadenza.