Progetto di reti Logiche

Interfaccia seriale asincrona UART

Davide Ghisolfi, Gioele Guaiumi

Indice

Introduzione

L'obiettivo del progetto è la progettazione di un'interfaccia seriale asincrona full duplex che contenga un trasmettitore ed un ricevitore UART dotati di segnalazione hardware del controllo di flusso. La trasmissione UART è basata sull'utilizzo di 4 linee digitali, due in ingresso e due in uscita:

- **TX**: trasmette il flusso informativo;
- **RX**: riceve il flusso informativo;
- **RTS** (Request To Send) e **CTS** (Clear To Send): quando il dispositivo ricevitore è pronto a ricevere una trasmissione, asserisce l'uscita RTS che; essendo collegata al segnale d'ingresso CTS del trasmettitore, ne comunica la disponibilità a ricevere; il segnale viene disasserito nel momento in cui il buffer di ricezione raggiunge la capacità massima.

Tutti i messaggi inviati da un trasmettitore hanno lunghezza di 10 bit, il primo e l'ultimo, detti *START* e *STOP* sono usati come delimitatori del messaggio, mentre i restanti otto costituiscono il contenuto informativo. Per iniziare la trasmissione, il trasmettitore trasmette lo START bit (bit basso), trasmette serialmente gli otto bit del messaggio e infine trasmette lo stop bit (bit alto) concludendo la trasmissione per poi mantenere la linea alta fino all'inizio di una nuova trasmissione.

La struttura del messaggio può essere di tre tipi:

- **8N1**: tutti gli otto bit portano contenuto informativo
- **7E1**: sette bit portano contenuto informativo, l'ottavo contiene la parità pari ed è pertanto vero solo se il numero di bit di contenuto uguali a 1 è pari
- **7O1**: sette bit portano contenuto informativo, l'ottavo contiene la parità dispari ed è pertanto vero solo se il numero di bit di contenuto uguali a 1 è dispari

START	D0	D1	D2	D3	D4	D5	D6	D7	STOP

START	D0	D1	D2	D3	D4	D5	D6	EVEN	STOP

START	D0	D1	D2	D3	D4	D5	D6	ODD	STOP

Dall'alto verso il basso le tre modalità di trasmissione: 8N1, 7E1 e 7O1

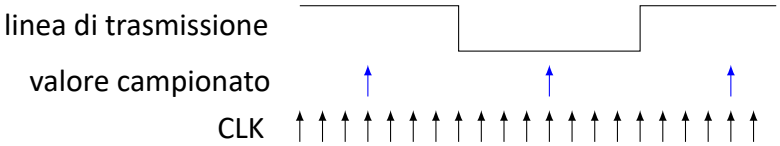
La modalità viene decisa dal trasmettitore al momento dell'invio attraverso due segnali **LEN** e **PARITY** che selezionano rispettivamente la lunghezza del contenuto informativo e il tipo di parità:

LEN	PARITY	Modalità
0	0	8N1
0	1	8N1
1	0	7E1
1	1	7O1

Nel momento in cui il dispositivo ricevente disasserisce il segnale *BC*, l'interfaccia che sta ricevendo lo comunica all'interfaccia che sta trasmettendo disasserendo il segnale *CTS*: il trasmettitore procede quindi a terminare l'eventuale trasmissione in corso per poi mantenere alta la linea di trasmissione.

Sovra-campionamento

Siccome la trasmissione è asincrona, sebbene i due dispositivi connessi funzionano alla stessa frequenza, non è garantito che i fronti dei due clock siano allineati. Inoltre la frequenza di trasmissione e ricezione è spesso affetta da errori e disturbi: per questi motivi il ricevente sovracampiona il segnale sulla porta RX ad una frequenza 8 volte più veloce della frequenza effettiva di funzionamento tenendo in memoria solamente il valore intermedio al fine di garantire una lettura corretta.



Frequenza di funzionamento

Le frequenze tipiche di funzionamento di un'interfaccia UART riportate in bit al secondo sono 9600, 19200, 38400, 57600, 115200, 230400, 460800 e 921600. Al fine di garantire un sovracampionamento in fase di ricezione, l'intera interfaccia deve essere alimentata da un segnale di clock ad una frequenza otto volte superiore alla frequenza di funzionamento.

Un ciclo di trasmissione dura sempre *10* cicli relativi alla frequenza di funzionamento (oppure *80* cicli relativi alla frequenza effettiva) a partire dal ciclo in cui viene asserito il segnale **START** (compreso): una nuova trasmissione può quindi cominciare dopo 10 cicli *di funzionamento* successivi all'asserimento del segnale START. Il comportamento dell'interfaccia è *indefinito* nel momento in cui il segnale START viene asserito mentre una trasmissione è già in corso ovvero dopo un intervallo minore del ciclo di trasmissione.

Specifica

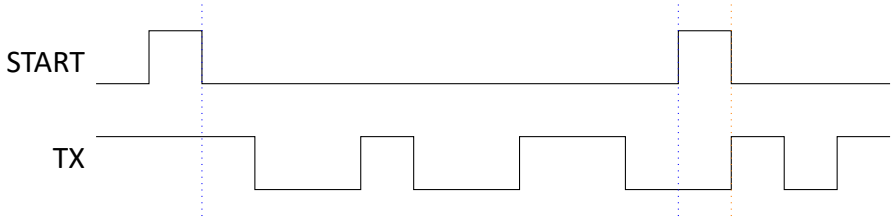
L'interfaccia è stata realizzata in modo da avere un modulo dedicato alla trasmissione e un modulo dedicato alla ricezione:

- **modulo di trasmissione TX**: riceve e memorizza in un registro *parallelo/serie* il dato da inviare dal bus di ingresso, sostituisce all'occorrenza l'ultimo bit del dato con il bit di parità e lo invia serialmente sulla linea quando il segnale START viene asserito se il segnale CTS è asserito;
- **modulo di ricezione RX**: riceve serialmente il dato dalla linea di comunicazione campionandolo ad una velocità otto volte superiore alla frequenza di funzionamento e lo memorizza in un registro *serie/parallelo*: quando la trasmissione si conclude, asserisce il segnale *READY* per un ciclo di clock di funzionamento.

Fase di trasmissione

In questa sezione il clock viene considerato come il segnale di sincronizzazione caratterizzato da una frequenza uguale alla frequenza di funzionamento in quanto ogni dispositivo all'interno del modulo di trasmissione lavora a tale frequenza.

Nel ciclo di clock in cui viene asserito il segnale **START** viene generato l'ultimo bit del messaggio, sia esso l'ottavo bit di DIN o il bit di parità. La fase di trasmissione inizia nel secondo ciclo di clock successivo con la trasmissione di un valore basso sulla linea di uscita. Negli otto cicli successivi la parola viene inviata serialmente sulla linea e al decimo ciclo la linea viene mantenuta alta fino all'inizio di una nuova trasmissione. Come riportato nella sezione introduttiva, un ciclo di trasmissione dura 10 cicli di clock.

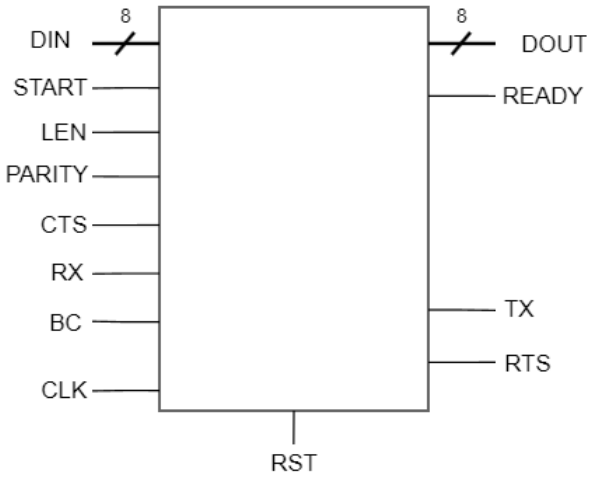


Esempio di trasmissione del byte 01001100 seguita da una trasmissione successiva

Fase di ricezione

Nel ciclo di clock in cui viene campionato dalla linea un segnale *RX_BUF* basso, il ricevitore abilita il contatore *CNT_SAMPLE* che permette di caricare il valore intermedio del bit ricevuto dalla linea di trasmissione contando il numero di bit ricevuti. Nel ciclo in cui viene ricevuto lo stop bit, viene asserito il segnale *READY* per un ciclo di clock di funzionamento.

Interfaccia del sistema



L'interfaccia prende in ingresso i seguenti segnali:

- il segnale **DIN** su 8 bit riceve il byte da inviare. È sufficiente che la parola rimanga sul bus solamente nel ciclo in cui *START* viene asserito;

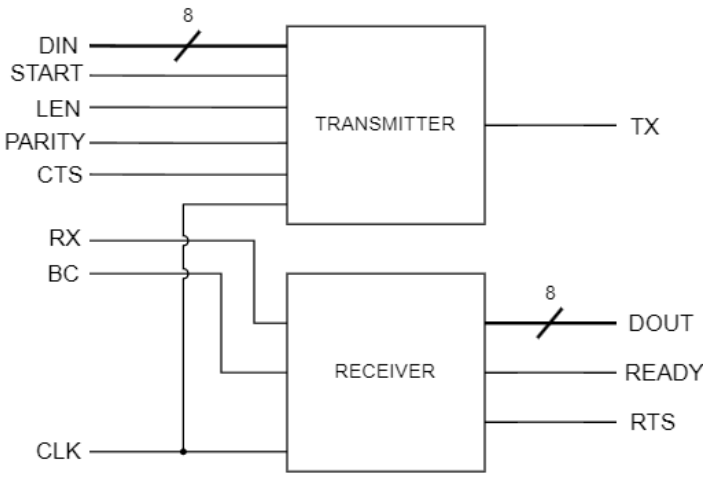
- il segnale digitale **START** viene asserito per *un ciclo* di clock quando si intende inviare il bit caricato su **DIN**;
- il segnale digitale **LEN** che vale 1 al momento dell'asserimento del segnale *START* se si intende inviare il dato in modalità 8N1. È sufficiente che il segnale resti asserito per un solo ciclo di clock;
- il segnale digitale **PARITY** che vale 1 al momento dell'asserimento del segnale *START* se si intende inviare il dato in modalità 7O1 oppure 0 se si vuole utilizzare la modalità 7E1. Come per il segnale *LEN* è sufficiente che il segnale resti asserito per un solo ciclo di clock;
- il segnale digitale **RX** su cui l'interfaccia riceve il segnale serializzato;
- il segnale digitale **CTS** vero se l'interfaccia ricevente è pronta a ricevere.

Ai segnali precedenti si aggiungono i segnali di reset **RST** e di clock **CLK** a cui il componente è sensibile solamente sul fronte di salita.

I segnali in uscita dell'interfaccia sono i seguenti:

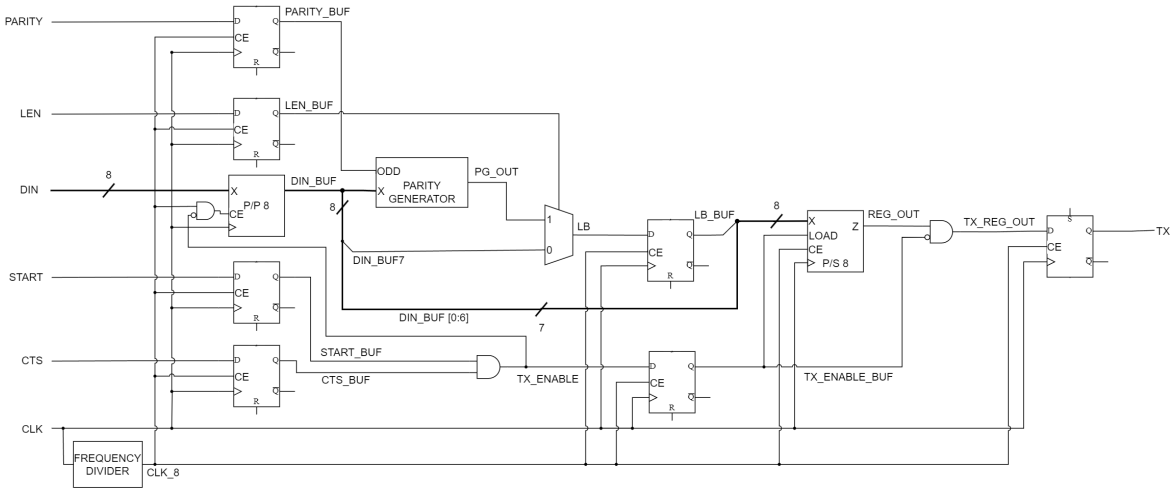
- il segnale **DOUT** su 8 bit su cui il modulo di ricezione rende disponibile il byte ricevuto;
- il segnale digitale **READY** che viene asserito dal modulo di ricezione per un ciclo di clock nel momento in cui si conclude la fase di ricezione;
- il segnale digitale **TX** usato dal modulo di trasmissione per inviare serialmente i bit del dato;
- il segnale digitale **RTS** asserito dal modulo di ricezione quando è pronto a ricevere e disasserito nel momento in cui il buffer del dispositivo ricevente è pieno.

Architettura del sistema



Il dispositivo è costituito da due moduli principali che lavorano indipendentemente al fine di garantire una connessione full duplex.

Trasmettitore



Nel ciclo (relativo alla frequenza di funzionamento) in cui il trasmettitore riceve il segnale *START*, supponendo che il segnale *CTS* sia asserito, i seguenti segnali vengono generati e il loro valore viene memorizzato in due flip flop D interni:

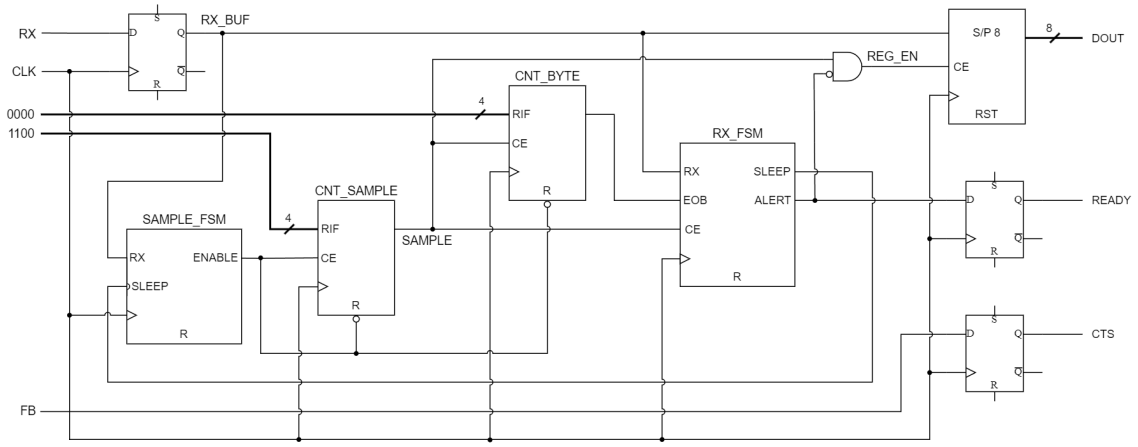
- **TX_ENABLE**: corrisponde all'AND logico tra *START* e *CTS*. La presenza di questa porta inibisce l'inizio della fase di trasmissione nel caso in cui il segnale *CTS* non sia asserito. Quando il segnale viene asserito disabilita inoltre il segnale clock enable (CE) del registro parallelo parallelo che funge da buffer per il segnale *DIN* mantenendolo in questo modo invariato durante il ciclo di clock successivo: questa operazione è necessaria in quanto non è noto il valore del segnale *DIN* nel momento in cui *START* viene disasserito.

- **LB** (Last Bit): rappresenta l'ultimo bit del messaggio da inviare. Viene generato partendo dai segnali **DIN_BUF7** (ultimo bit di DIN_BUF), **LEN_BUF** e **PAR_BUF** che rispettivamente rappresentano i buffer dei segnali *DIN7*, *LEN* e *PARITY*: se *LEN_BUF* è basso, viene selezionato il segnale **DIN_BUF7**, altrimenti viene selezionata l'uscita del *PARITY GENERATOR*.

modalità	LEN	PARITY	LB
8N1	0	-	DIN_BUF7
7E1	1	0	PG_OUT
7O1	1	1	PG_OUT

Nel secondo ciclo il segnale *TX_ENABLE_BUF* collegato al segnale *LOAD* del registro parallelo serie permette il caricamento della parola di 8 bit composta dai primi 7 bit di *DIN_BUF* concatenati a *LB_BUF* nel registro parallelo serie e impone sull'uscita del flip flop finale che ha come uscita il segnale *TX* un segnale basso: in questo modo all'inizio del ciclo successivo il segnale *TX* sarà basso e inizierà la trasmissione. Negli otto cicli successivi, siccome il segnale *START* è basso, anche il segnale *TX_ENABLE_BUF* si abbassa e permette ai bit d'uscita del registro parallelo serie di arrivare all'ingresso dell'ultimo flip flop e di conseguenza di essere trasmessi sulla linea. In questa fase il registro parallelo serie si comporta come un registro a scorrimento facendo scorrere ad ogni ciclo di clock i valori dal bistabile i al bistabile $i+1$ e inserendo nel primo un valore *alto*: in questo modo al termine della trasmissione e fino all'inizio di una trasmissione successiva il registro impone sulla linea di uscita un valore alto.

Ricevitore



Di seguito sono brevemente descritti i principali segnali del ricevitore al fine di fornire un riferimento che faciliti la descrizione dell'architettura:

- **CNT_SAMPLE**: contatore a 8 bit usato come segnale di clock enable dei componenti a valle (*CNT_BYTE*, *RX_FSM* e *S/P_8*) soltanto per il ciclo intermedio al periodo di trasmissione di ogni bit. Il suo riferimento (*RIF*) è 1100, ossia il secondo simbolo della sequenza del contatore in modo tale da abilitare i componenti a monte solamente nel ciclo di clock mediano alla frequenza di funzionamento, ovvero il quarto ciclo di clock di alimentazione a partire dal fronte di salita del clock di funzionamento.

- **CNT_BYTE**: contatore a 8 bit che ha lo scopo di contare i bit ricevuti e avvisare *RX_FSM* dell'avvenuta ricezione dell'intero byte. Per questo motivo il suo riferimento è impostato a 0000, l'ultimo simbolo della sequenza del contatore.

- **RX_FSM**: macchina a stati finiti che descrive lo stato del ricevitore e controlla la fase finale della ricezione. La sua uscita *ALERT* avvisa l'utente che il byte ricevuto è pronto per essere letto e azzera i contatori al termine della trasmissione.

- **SAMPLE_FSM**: macchina a stati finiti che abilita il contatore *CNT_SAMPLE* e resetta i contatori quando finisce la trasmissione

- **S/P_8**: registro serie/parallelo a 8 bit. È abilitato dal segnale *REG_EN* che rappresenta l'AND logico tra il segnale *SAMPLE* e il segnale *ALERT* negato in modo che sia opaco solamente se *SAMPLE* è asserito e la macchina non è in stato di *ALERT*.

La trasmissione del dato comincia nel momento in cui il segnale *RX_BUF*, campionamento dell'ingresso *RX*, si abbassa. Al ciclo successivo la macchina a stati *SAMPLE_FSM* passa in stato ENABLE alzando il segnale *SAMPLE_EN* e abilitando di conseguenza il *CNT_SAMPLE*, la cui uscita (*SAMPLE*) funge da Clock Enable (CE) per i componenti a valle, portandoli a lavorare ad un ottavo della frequenza d'ingresso *CLK* (frequenza di funzionamento). Negli otto cicli successivi il segnale *RX_BUF* viene caricato nel registro *serie/parallelo* e viene incrementato il valore del contatore *CNT_BYTE*. Una volta che l'intero byte viene ricevuto, il segnale *EOB* (End Of Byte) si alza e *RX_FSM* passa in stato di ALERT che asserisce il segnale di uscita *ALERT* il quale inibisce il caricamento di segnali nel registro. Nel ciclo successivo *RX_FSM* torna nello stato di attesa (*WAIT*), abbassa il segnale di *READY* e porta *SAMPLE_FSM* in stato WAIT: di conseguenza i contatori *CNT_SAMPLE* e *CNT_BYTE* vengono

resettati; in questo modo il ricevitore è tornato allo stato iniziale ed è pronto a ricevere il messaggio successivo.

Moduli base

MUX (Multiplexer)

Il *multiplexer* è un circuito che permette di selezionare da un insieme di segnali X in ingresso un segnale Z da inviare all'uscita mediante un altro segnale in ingresso S definito *selettore*. Di seguito si riporta la formula in forma SoP di un multiplexer a due ingressi:

$$Z = X_0 \cdot \overline{S} + X_1 \cdot S$$

Flip flop

Il flip flop è un circuito sequenziale che permette di memorizzare un bit di informazione. Lo stato del flip flop è sensibile solamente al fronte di salita del clock e può pertanto (idealmente) cambiare stato solamente nell'istante in cui tale segnale passa da stato basso a stato alto. A seconda della modalità con cui viene imposto lo stato si distinguono vari tipi di flip flop:

- **flip flop JK**: è dotato di due ingressi J e K che rispettivamente portano lo stato ad alto e basso quando sono asseriti. Nel momento in cui entrambi non sono asseriti il componente mantiene lo stato mentre se entrambi sono asseriti lo stato prossimo è lo stato corrente negato

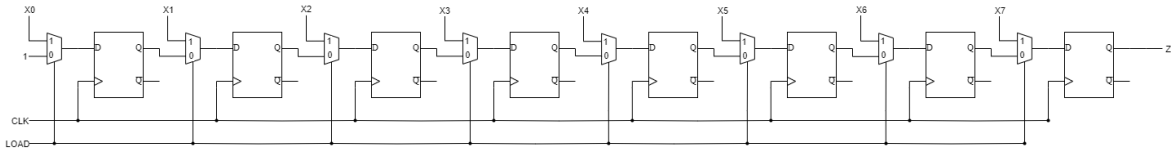
J	K	Q^*
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

- **flip flop D**: è dotato di un solo ingresso D che corrisponde allo stato prossimo del flip flop.

Entrambi i componenti sono dotati di un segnale di reset *asincrono* che se asserito mantiene lo stato basso e i flip flop D sono anche dotati di un segnale SET sempre asincrono che se asserito mantiene lo stato alto.

Registro P/S (parallelo serie)

Un registro è un insieme di flip-flop che permette di memorizzare parole di dimensione maggiore del bit. Il registro *parallelo serie* su n bit è un tipo di registro dotato di un ingresso a n bit X, di un ingresso *LOAD* e un'uscita seriale Z. Nel momento in cui il segnale *LOAD* viene asserito, la parola presente su X viene caricata nei flip flop. Se il segnale *LOAD* non è asserito, invece, il componente si comporta come un registro a scorrimento e trasferisce il bit del flip flop i-esimo sul flip flop $(i+1)$-esimo mentre impone all'ingresso del primo flip flop un segnale *alto*.

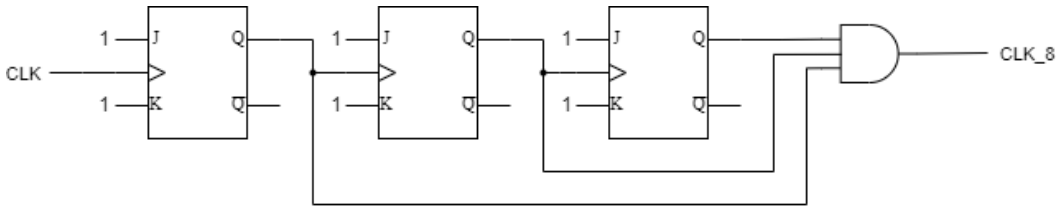


Registro S/P (serie parallelo)

Il registro *parallelo serie* a n bit è un registro dotato di un ingresso seriale X e di un'uscita a n bit Y. Permette di caricare i valori in maniera seriale e di leggere il valore contenuto in parallelo leggendo il valore di uscita di ogni bistabile.

Frequency divider

Il *frequency divider* è un circuito sequenziale che prende in input un segnale periodico di periodo T e genera sull'uscita un segnale di periodo $2^N T$ e duty cycle pari al 25% con N numero intero positivo. È composto da N flip flop JK collegati in modo tale che entrambi gli ingressi siano asseriti e il flip flop i-esimo riceva come clock in ingresso l'uscita del flip flop $i-1$-esimo Q_{i-1}: l'uscita dell'i-esimo flip flop Q_i ha periodo pari a $2^i T$. Inoltre l'uscita del circuito corrisponde all'AND logico delle uscite di ogni flip flop in modo che il componente restituisca un segnale con il duty cycle cercato. Il duty cycle indicato è necessario in quanto all'interno della rete il segnale d'uscita viene impiegato come segnale clock enable: in questo modo il componente attivato da *CLK_8* è opaco ad un solo ciclo del clock veloce.

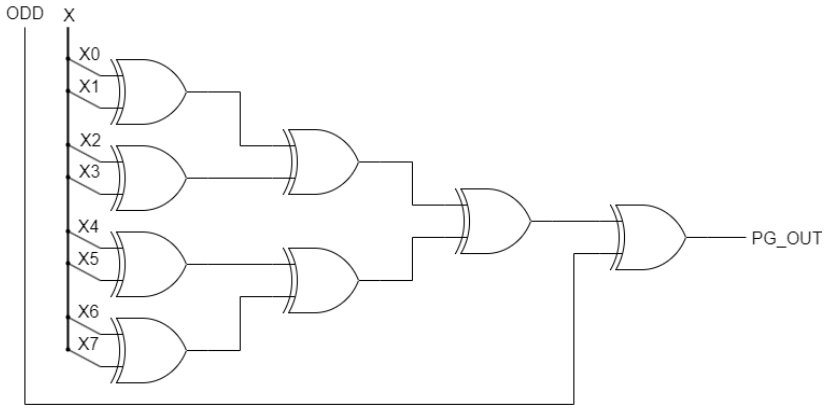


Parity generator

Il *parity generator* è un circuito combinatorio dotato di un ingresso D a n bit, un ingresso binario *ODD* e una uscita *PG_OUT* che genera la parità di D. In maniera più formale, l'uscita del componente è vera se e solo se la *cardinalità* di D (ovvero il numero di linee asserite) è un numero dispari e *ODD* non è asserito oppure se la *cardinalità* di D è un numero pari e *ODD* è asserito.

$$PG\_OUT = \|D\| \oplus ODD$$

$\|D\|$	ODD	PG_OUT
PARI	0	0
PARI	1	1
DISPARI	0	1
DISPARI	1	0



RX_FSM

La macchina a stati del ricevitore ha il compito di segnalare l'avvenuta ricezione del byte inviato e di controllare il termine della trasmissione.

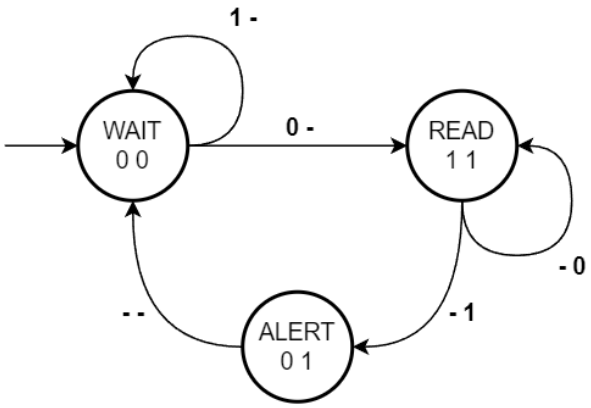


Diagramma di RX_FSM con ingressi e uscite nell'ordine RX EOB, SLEEP ALERT

Stato WAIT (W) stato iniziale e di default della macchina. In questo stato il ricevitore è in attesa di una nuova trasmissione. Nel momento in cui il segnale *RX* si abbassa, la macchina passa nello stato **READ**;

Stato READ (R) il ricevitore legge e salva i bit campionati nel registro parallelo/serie. Quando l'ingresso *EOB* si alza segnalando l'avvenuta lettura del byte, la macchina passa nello stato **ALERT**;

Stato ALERT (A) segnala l'avvenuta ricezione del byte asserendo il segnale **ALERT**. Nel ciclo di clock successivo il ricevitore ritorna nello stato di **WAIT** in attesa della prossima trasmissione.

Segnale	Direzione	Tipo	Descrizione
RST	IN	BIT	Segnale di reset del circuito. Finché asserito alto, lo stato *W* rimane impostato
CLK	IN	BIT	Segnale di sincronizzazione (sul fronte di salita)
CE	IN	BIT	Quando asserito alto abilita *CLK*
RX	IN	BIT	Segnale (campionato) ricevuto dal trasmettitore
EOB	IN	BIT	End Of Byte : avverte la completa ricezione del byte
ALERT	OUT	BIT	Asserito alto nello stato di *ALERT* della FSM
SLEEP	OUT	BIT	Asserito alto nello stato di *WAIT* della FSM

tabella IO di RX_FSM

SAMPLE_FSM

La macchina a stati del campionamento ha il compito di abilitare il contatore di sample e di resettare i due contatori una volta che la trasmissione si conclude

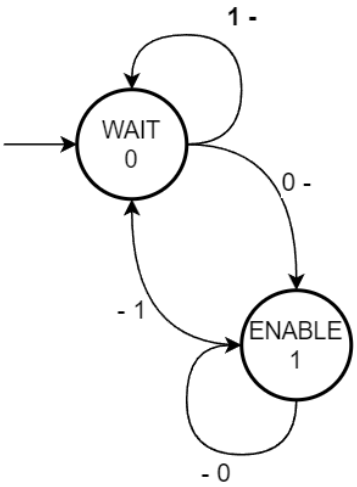


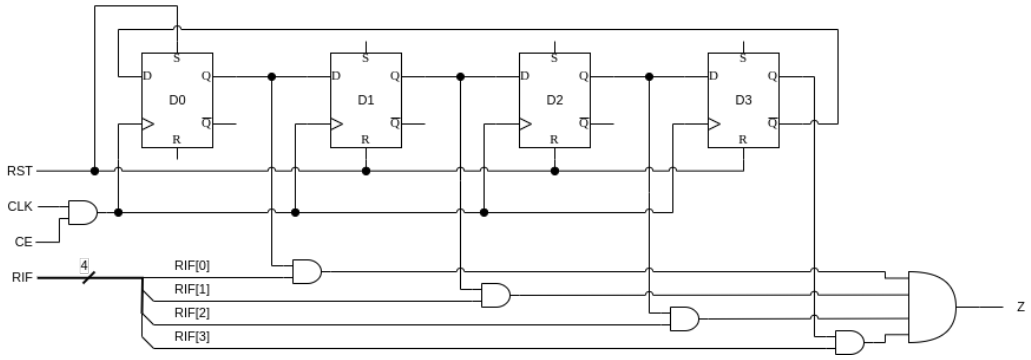
Diagramma di SAMPLE_FSM con ingressi e uscite nell'ordine RX SLEEP, ENABLE

Stato WAIT (W) stato iniziale e di default della macchina. In questo stato il segnale è in attesa di una nuova trasmissione. Nel momento in cui il segnale *RX* si abbassa, passa nello stato *ENABLE*;

Stato ENABLE (E) nello stato ENABLE la macchina ha uscita alta. Rimane in questo stato finché la macchina a stati *RX_FSM* non entra in stato *SLEEP*. A questo punto torna in stato *WAIT*.

Contatore a 8 bit

Il contatore a 8 bit è un circuito sequenziale basato sul contatore di Moebius dotato di un ingresso **RIF** su 4 bit e di un uscita binaria Z vera se e solo se la sequenza del contatore è uguale al segnale RIF.



Il contatore segue ciclicamente la sequenza di stati fornita di seguito:

$$1000,\ 1100,\ 1110,\ 1111,\ 0111,\ 0011,\ 0001,\ 0000$$

Segnale	Direzione	Tipo	Descrizione
RST	IN	BIT	Segnale di reset del circuito. Finché asserito mantiene lo stato 1000
CLK	IN	BIT	Segnale di sincronizzazione Il componente è sensibile sul fronte di salita
CE	IN	BIT	Quando asserito alto permette a *CLK* di rendere il componente opaco
RIF	IN	VEC	Vettore di 4 bit, specifica lo stato del contatore nel quale l'uscita è alta
Z	OUT	BIT	Vero se il contatore si trova nello stato corrispondente a *RIF*

tabella IO del contatore

Verifica

Il test bench è stato realizzato in modo da evidenziare la capacità del componente di lavorare in condizioni in cui i clock dei due componenti connessi abbiano frequenze di funzionamento leggermente differenti (nell'intorno del 2%) e con fronti non allineati. Siccome l'interfaccia può lavorare ad una serie di frequenze fissate si è deciso di eseguire i test alla frequenza di funzionamento massima, ovvero 921600 b/s: di conseguenza la frequenza di clock con cui viene alimentato il componente è pari a 7372800Hz ($T \simeq 135ns$). Si è deciso inoltre di sfasare il clock del dispositivo ricevitore di metà periodo e di aumentare la sua frequenza di due punti percentuale. Le funzioni dei due clock possono quindi essere messe in relazione secondo la seguente uguaglianza:

$$CLK_R(t) = CLK_T\left(1.02\,t + \frac{T_T}{2}\right)$$

con $CLK_T(t)$ segnale di clock del trasmettitore, T_T periodo del clock del trasmettitore e CLK_R segnale di clock del ricevitore.

I test effettuati sono volti sia a verificare la corretta ricezione del messaggio in presenza di frequenze diverse sia a verificare l'effettivo comportamento del trasmettitore al variare dei segnali di lunghezza, parità e CTS:

1. **Test messaggio tipo 8N1**
    - LEN = 0
    - PARITY = 0
    - **DIN** = 10110101
    - **DOUT** = 10110101
2. **Test messaggio tipo 7E1**
    - LEN = 1
    - PARITY = 0
    - **DIN** = 00011100
    - **DOUT** = 00011101
3. **Test messaggio tipo 7O1**
    - LEN = 1
    - PARITY = 1
    - **DIN** = 01001110
    - **DOUT** = 01001111
4. **Test disasserimento CTS durante la trasmissione**
    - LEN = 0
    - PARITY = 0
    - **DIN** = 10101010
    - **DOUT** = 10101010

    durante il test il segnale CTS viene disasserito per un intervallo di tempo superiore ad un ciclo di trasmissione e viene asserito il segnale di START (per un ciclo di clock).

Il test complessivo è composto dall'invio dei seguenti messaggi:

1. Test messaggio tipo 8N1
2. Test messaggio tipo 7E1
3. Test messaggio tipo 7O1
4. Test disasserimento CTS durante la trasmissione
5. Invio di 1 messaggio generico