Esercizio 1

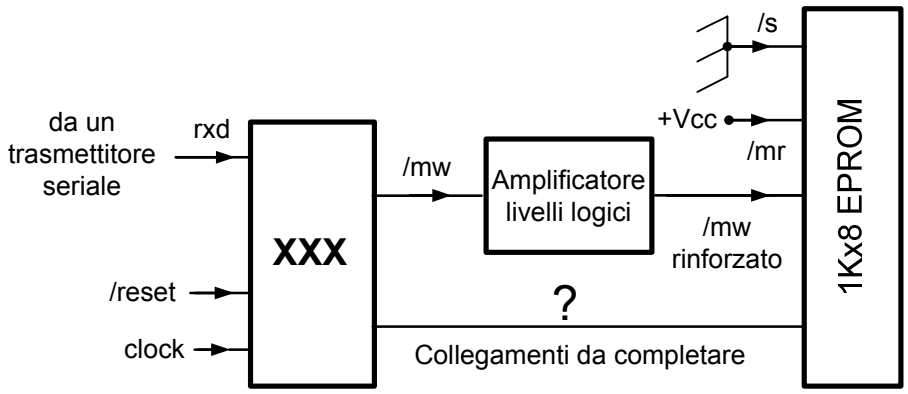
La figura rappresenta un circuito di *estensione di campo* per interi rappresentati in traslazione in una base generica β (pari).

1) Trovare la relazione algebrica che lega l'uscita all'ingresso.
2) Sintetizzare il circuito che esegue l'operazione di estensione, inserendo meno logica possibile, in base generica β.
3) Sintetizzare lo stesso circuito in base 2. Il circuito risultante ha complessità nulla?

Esercizio 2

L'Unità *XXX* è, grazie alla presenza dell'amplificatore del segnale */mw*, un programmatore di EPROM capace quindi di scrivere nella EPROM.

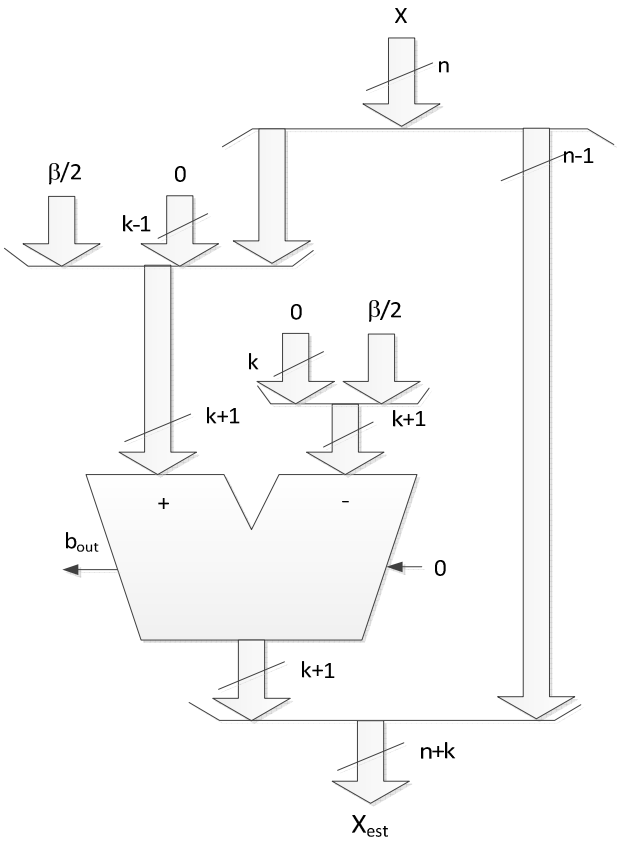
L'Unità riceve da un trasmettitore seriale, secondo lo standard START/STOP, trame con 18 bit utili e utilizza ognuna di esse per una operazione di scrittura nella EPROM.



1) Specificare i collegamenti indicati *come da completare* e indicare preliminarmente come verranno usati (nella descrizione Verilog di *XXX*) i 18 bit utili delle trame.
2) Descrivere in Verilog l'Unità *XXX* supponendo che:
   - Il suo clock abbia **un periodo 8 volte inferiore** al tempo di bit della comunicazione seriale asincrona START/STOP
   - Un ciclo di scrittura nella EPROM sia del tutto simile ad un ciclo di scrittura in una RAM
   - L'intervallo tra l'arrivo di due trame successive sia talmente lungo da non creare problemi di nessun tipo.
3) Sintetizzare la (sola) porzione della Parte Operativa relativa al registro BUFFER in cui vengono raccolti i 18 bit utili delle trame.

Esercizio 1 – soluzione

1) La relazione che lega un intero x alla sua rappresentazione in traslazione su n cifre X è $X = x + \beta^n/2$, $-\beta^n/2 \le x \le \beta^n/2 - 1$. Dalla precedente si ottiene $X_{est} = x + \beta^{n+k}/2 = X - \beta^n/2 + \beta^{n+k}/2$.
In base due, la relazione diventa $X_{est} = X - 2^{n-1} + 2^{n+k-1}$



2)
$X_{est} = X + \beta^{n+k}/2 - \beta^n/2 = X + \beta^{n+k-1} \cdot \beta/2 - \beta^{n-1} \cdot \beta/2$.
La prima delle due operazioni è un concatenamento, e quindi non necessita di logica. La seconda è una sottrazione *a k*+1 *cifre*, le cifre più alte di X_{est}, come da figura sottostante.

k+1					n-1				
$\beta/2$	0	...	0	x_{n-1}	x_{n-2}	...	x_2	x_1	x_0
0	0	...	0	$\beta/2$	0	...	0	0	0

Il circuito che realizza l'operazione è il seguente:

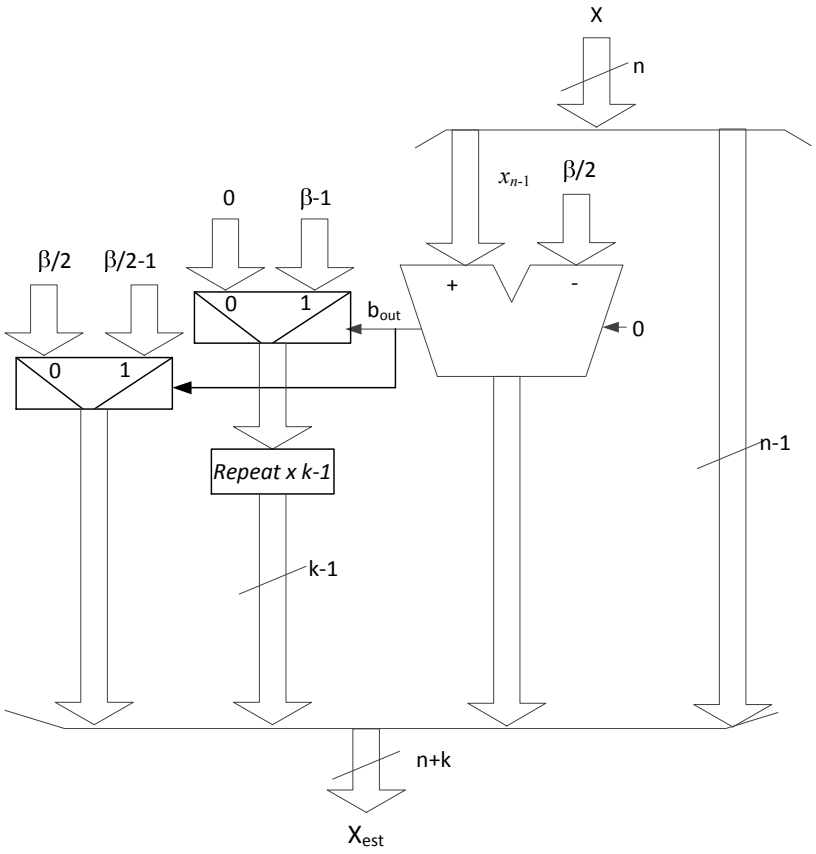
Un circuito ulteriormente ottimizzato si ricava osservando che le k cifre più significative di X_{est} possono avere soltanto due configurazioni:

- $\beta/2, 0, 0, .., 0$ (se $x_{n-1} \ge \beta/2$)
- $\beta/2 - 1, \beta - 1, \beta - 1, ..., \beta - 1$ (se $x_{n-1} < \beta/2$)

Pertanto, si può sostituire il sottrattore a k+1 cifre con:

a) Un sottrattore ad una cifra, che esegue la sottrazione $x_{n-1} - \beta/2$, e produce x_{n-1}^{est} e b_{out}.
b) Due multiplexer con ingressi costanti, guidati da b_{out}. Il primo multiplexer ha in ingresso $\beta/2$ e $\beta/2 - 1$, il secondo 0 e $\beta - 1$. Le uscite di questi multiplexer danno le k cifre più significative del risultato.

Il circuito così ottenuto ha meno logica del precedente ed è più veloce, in quanto non ha la catena dei riporti.

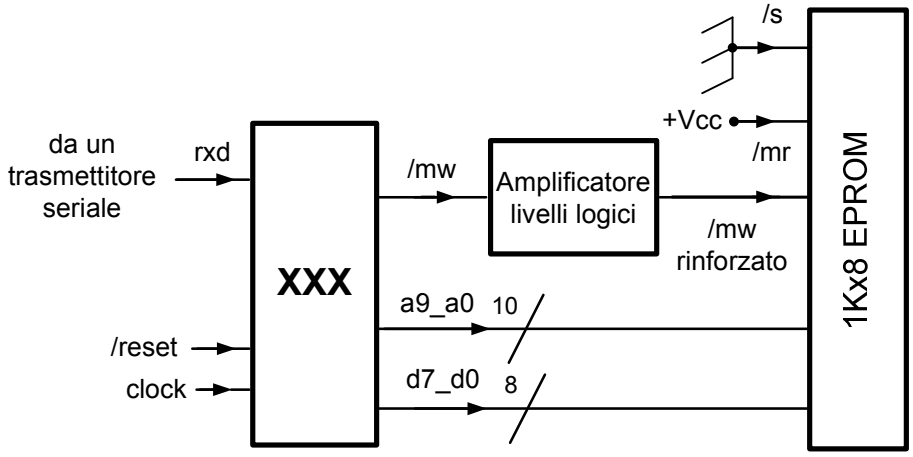


3) In base 2 lo schema può essere semplificato, considerando che $\beta/2 = 1$. Infatti, il risultato della sottrazione su k+1 cifre è:
- 011…11, se $x_{n-1} = 0$
- 100…00, altrimenti.

Pertanto, la sottrazione può essere sostituita dalla *giustapposizione* dei bit $x_{n-1}, \overline{x_{n-1}}, \overline{x_{n-1}}, \overline{x_{n-1}}, ..., \overline{x_{n-1}}, \overline{x_{n-1}}$.
Il circuito risultante è quindi di complessità (quasi) nulla – "quasi" per via dell'invertitore.
Allo stesso risultato si arriva osservando che la rappresentazione in traslazione in base 2 si ottiene da quella in complemento alla radice complementando il bit più significativo. Pertanto, si può convertire la rappresentazione in CR, estenderla, e riconvertirla in traslazione.

Esercizio 2 - soluzione

Si presuppone che i bit vengano spediti (e quindi ricevuti) nel seguente ordine:
bit di start, a0, a1,…., a9, d0, d1, …., d7, bit di stop
Se si assegna
`BUFFER<={rxd,BUFFER[17:1]}`
allora i *dati* verranno a occupare, all'interno del BUFFER, le 8 posizioni più significative e gli *indirizzi* le 10 posizioni meno significative




```
module XXX(d7_d0, a9_a0, mw_, rxd, clock, reset_);
 input        clock, reset_;
 input        rxd;
 output[7:0]  d7_d0;
 output[9:0]  a9_a0;
 output       mw_;
 reg       MW_;    assign mw_=MW_;
 reg[4:0]  COUNT;
 reg[3:0]  WAIT;
 reg[17:0] BUFFER; assign d7_d0=BUFFER[17:10]; assign a9_a0=BUFFER[9:0];
 reg [2:0] STAR;   parameter S0=0, S1=1, S2=2, S3=3, Wbit=4,Wstop=5;
 parameter start_bit=1'B0;
 always @(reset_==0) begin MW_=1; STAR<=S0; end
 always @(posedge clock) if (reset_==1)
  casex (STAR)
   S0: begin COUNT<=18; WAIT<=11; STAR<=(rxd==start_bit)? Wbit:S0; end
   S1: begin BUFFER<={rxd,BUFFER[17:1]}; COUNT<=COUNT-1;
             WAIT<=7; STAR<=(COUNT==1)?Wstop:Wbit; end
   S2: begin MW_<=0; STAR<=S3; end
   S3: begin MW_<=1; STAR<=S0; end
   Wbit:    begin WAIT<= WAIT-1; STAR<=(WAIT==1)?S1:Wbit; end
   Wstop:   begin WAIT<= WAIT-1; STAR<=(WAIT==1)?S2:Wstop; end
  endcase
endmodule
```