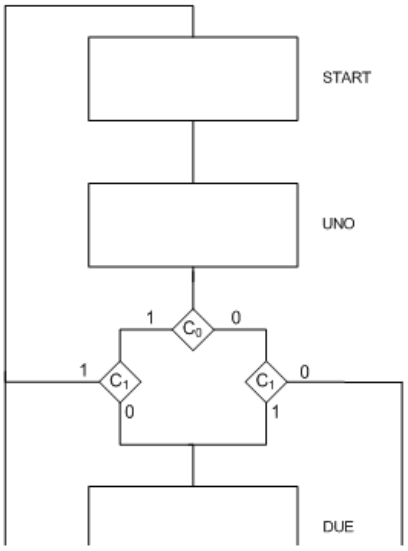
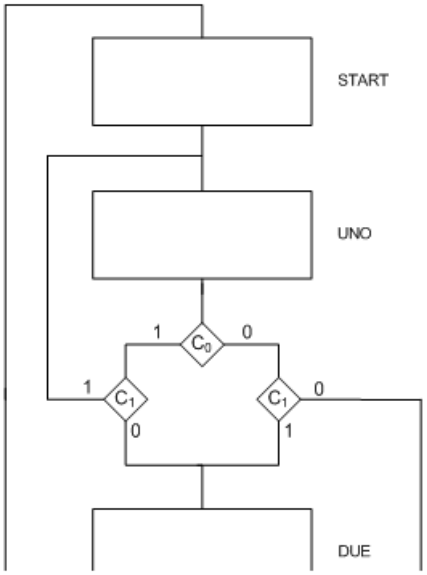
Elementi di progettazione dei sistemi VLSI
Volume III: Esercizi di progetto
ERRATA CORRIGE E NOTE AGLI ESERCIZI

- Esercizio n. 12 - pag. 19, diagramma ASM:

Errata:



Corrige:



- Esercizio n. 13 - pag. 24, 3[a] riga dal basso:
  Errata: "[...] "101110 [...]"
  Corrige: "[...] "10110 [...]"

- Esercizio n. 13 - pag. 26, prima riga, seconda colonna della 2[a] tabella fondo pagina:

Errata:

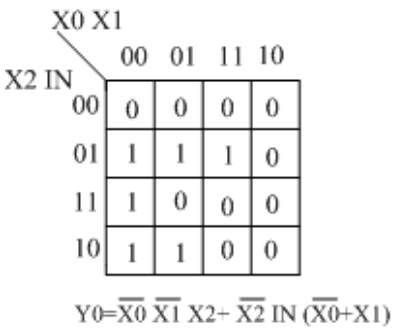
IN	X1X2X0	Y2Y1Y0	ALARM

Corrige:

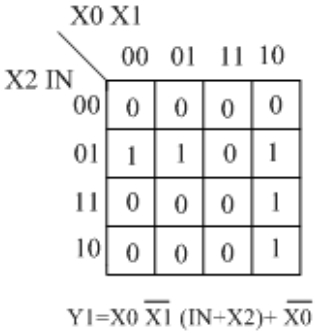
IN	X2X1X0	Y2Y1Y0	ALARM

- Esercizio n. 13 - pag. 27, prime due mappe di Karnaugh:

Errata:

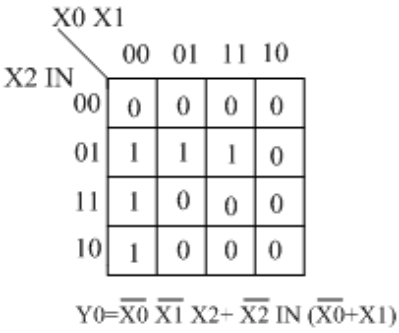


$Y0=\overline{X0}\ \overline{X1}\ X2+\overline{X2}\ IN\ (\overline{X0}+X1)$

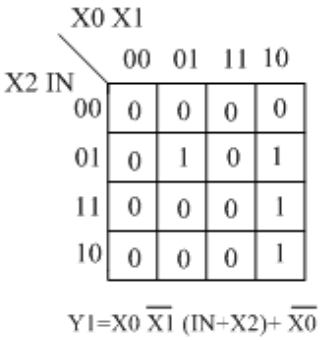


$Y1=X0\ \overline{X1}\ (IN+X2)+\overline{X0}\ X1\ \overline{X2}\ IN$

Corrige:



$Y0=\overline{X0}\ \overline{X1}\ X2+\overline{X2}\ IN\ (\overline{X0}+X1)$



$Y1=X0\ \overline{X1}\ (IN+X2)+\overline{X0}\ X1\ \overline{X2}\ IN$

- Esercizio n. 15 - pag. 31, 4[a] riga dall'alto: *Errata sintesi di Y1:*

Errata:

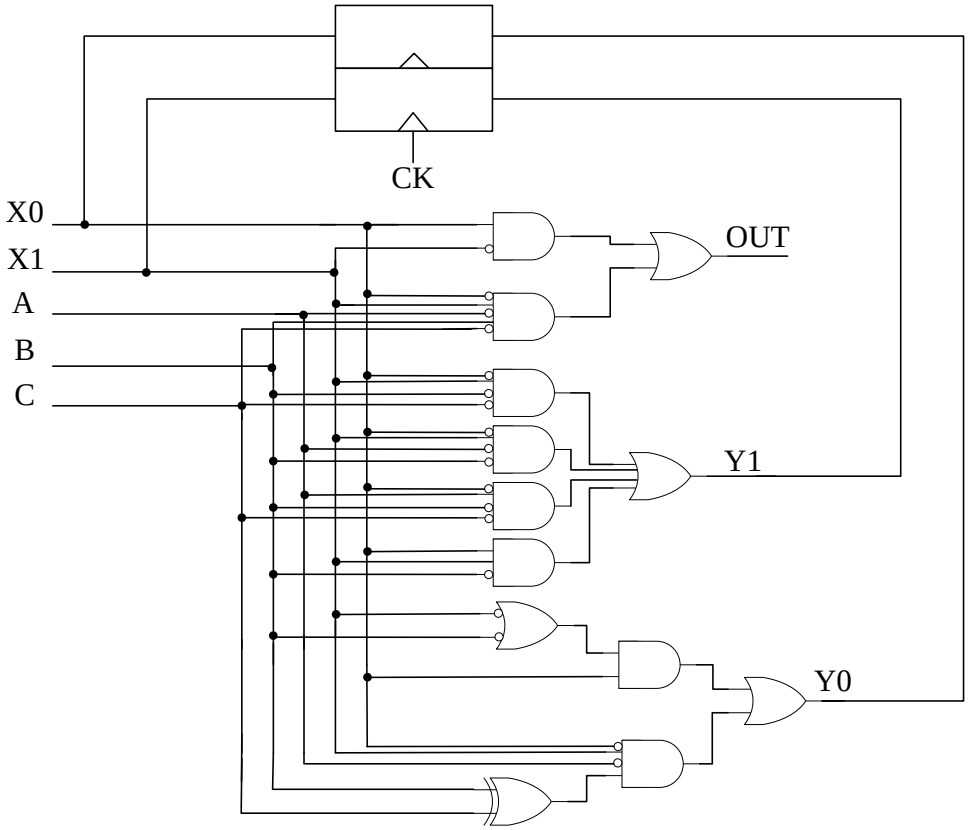
$$Y1 = \bar{B}\bar{C}\ \overline{X0}X1 + \bar{A}\bar{B}\overline{X0}X1 + A\bar{B}C\overline{X0} + \bar{B}X0X1$$

Corrige:

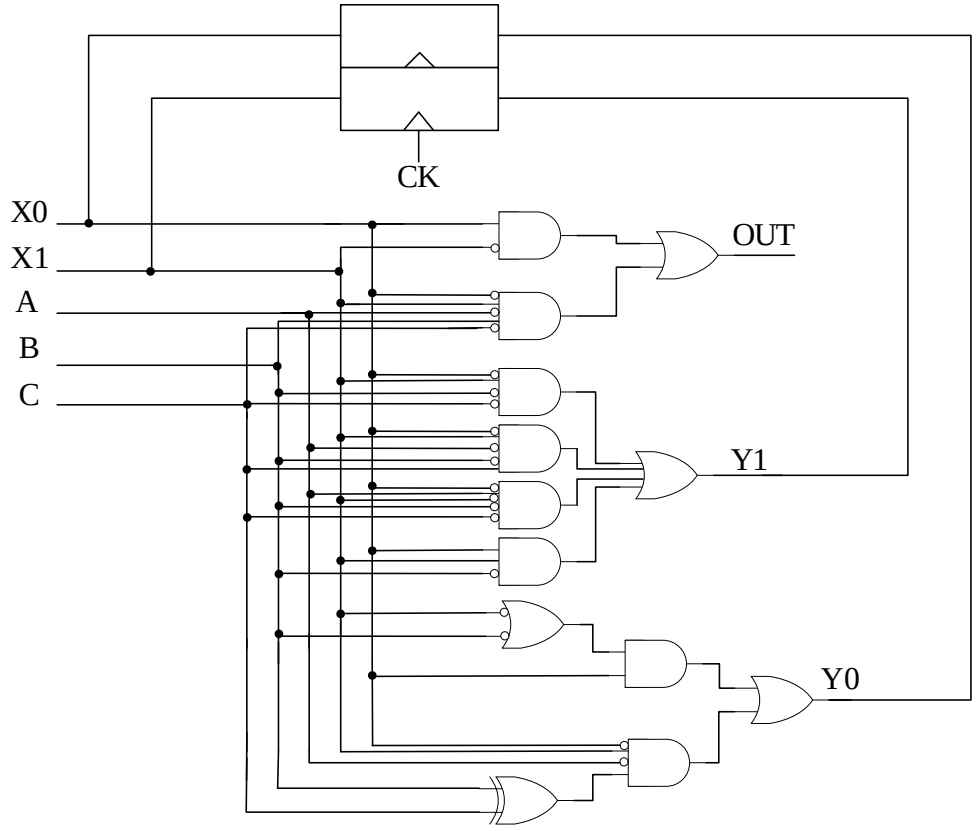
$$Y1 = \bar{B}\bar{C}\ \overline{X0}X1 + \bar{A}\bar{B}\overline{X0}X1C + A\bar{B}\bar{C}\ \overline{X0}\ \overline{X1} + \bar{B}X0X1$$

- Esercizio n. 15 - pag. 31, figura: *errati collegamenti inerenti la sintesi di Y1(si veda la correzione precente):*

Errata:



Corrige:



- Esercizio n. 16 - pag. 32, 3ª riga dopo la figura:

  Errata: "[...] fare uno shit [...]"
  Corrige: "[...] fare uno shift [...]"

- Esercizio n. 18 - pag. 38, 3ª riga dopo la tavola di trasizione degli stati: *frase da eliminare:*

  Errata: "Imponendo J=Y0 e K=Y1 si ottiene,"
  Corrige: Eliminare la frase "Imponendo J=Y0 e K=Y1 si ottiene,"

- Esercizio n. 38 - pagg. 104 e 105 prima del punto e virgola in ogni descrizione*: condizione di default mancante:*

  Errata: `WHEN a='1' AND B='1' ;`
  Corrige: `WHEN a='1' AND B='1' ELSE '0' ;`

- Esercizio n. 46 - pag. 113, 8ª riga dall'alto*: dichiarazione di un segnale non usato:*

  Errata: `SIGNAL state : std_logic ;`
  Corrige: Di per se non costituisce nessun errore ma un segnale dichiarato e non usato è segnalato per mezzo di un warning da tutti i più comuni sintetizzatori. In genere è buona norma dichiarare solo i segnali effettivamente usati.

- Esercizio n. 52 - pag. 121, 14ma riga dall'alto: *la dichiarazione di un processo non deve finire con la parola riservata "IS":*

  Errata: `ControlPath: PROCESS (CK) IS`
  Corrige: `ControlPath: PROCESS (CK)`

- Esercizio n. 54 - pag. 127, diagramma ASM: *test dell'ingresso nell'ultimo stato:*

  Errata: Dallo stato "101" si va direttamente allo stato "001"
  Corrige: Nello stato "101" bisogna aggiungere una condizione di test (rombo) sul segnale input: se '1' la FSM deve portarsi in "010" (il primo bit di una nuova sequenza è pari a uno) altrimenti in "001"

- Esercizio n. 54 - pag. 127, diagramma ASM: *condizioni mancanti nel terzo test dell'ingresso:*

  Errata: Il rombo non contiene le indicazioni del valore dell'ingresso secondo cui viene presa la decisione sul nextstate
  Corrige: Il rombo e le indicazioni sulle frecce devono essere come nello stato precedente

- Esercizio n. 54 - pag. 128, 13ma riga dal basso: *test dell'ingresso nell'ultimo stato:*

Errata:
```
case 4 : { nextstate = 0 ;
           Alarm = 1 ;
         }
         Break ;
```
Corrige:
```
case 4 : if (input==0)
           nextstate = 0 ;
         else
           nextstate = 1 ;
         Alarm = 1 ;
         Break ;
```

- Esercizio n. 54 - pag. 128, codice C: *assegnazione del valore Alarm:*

Errata: Nel caso 0 e nel caso 1 non viene assegnato alcun valore alla variabile Alarm

Corrige: Nel caso 0 e nel caso 1 deve essere assegnato alla variabile Alarm il valore 0 così che Alarm ritorni a 0 quando dal caso 4, in cui Alarm assume valore 1, si va nel caso 0 o nel caso 1

- Esercizio n. 54 - pag. 130, 15ma riga dall'alto: *test dell'ingresso nell'ultimo stato:*

Errata:
```
nextstate <= "001";
```
Corrige:
```
if input = '1' then
    nexstate <= "010";
else
    nexstate <= "001";
end if;
```

- Esercizio n. 58 - pag. 137, 11ma riga dal basso: *errore di battitura:*

Errata: "Di seguito riportiamo le assegnazioni di segnali per a ciascuno stato."

Corrige: "Di seguito riportiamo le assegnazioni di segnali per ciascuno stato."

- Esercizio n. 58 - pag. 137. *Nota preliminare: come in ogni progetto, un progettista deve operare delle scelte (ad esempio, in un Mux 2:1 quale dei due ingressi portare in uscita quando il selettore vale 0) con l'accortezza di attenersi a queste nell'intero progetto. Nello svolgimento del progetto proposto nel testo, si è implicitamente scelto che:*
    - *qualora i selettori dei MUX all'ingresso dei registri Read e Write (Sel_Mux_Read e Sel_Mux_Write) valgano 1 allora le uscite dei MUX assumono il valore del registro 1, viceversa se valgono 0 le uscite assumono il valore del registro 0;*
    - *qualora i selettori dei MUX all'ingresso dei registri i e j (Sel_Mux_i e Sel_Mux_j) valgano 1 allora le uscite dei MUX assumono il valore contenuto sul bus 3, viceversa se valgono 0 le uscite assumono il valore del registro 0;*

- *i segnali Cmd_Alu_Sub_b1_b2 e Cmd_Alu_Sub_b2_b1 attivano (quando sono posti a '1') le operazioni di sottrazione bus2-bus1 e bus1-bus2 rispettivamente;*
- *infine si assume che l'uscita del comparatore (che ha come ingressi il bus1 e il bus3) valga 1 se bus1>bus3 e 0 nell'altro caso.*

Sulla base di quanto appena detto si evidenziano i seguenti errori nello svolgimento dell'esercizio:

STATO 5, pag. 139: errore di battitura nella spiegazione:

Errata:

Reg_1_to_bus2 <= '1'	Mette in bassa impedenza il **tristate-buffer** connesso tra l'uscita del registro **1** e il **bus2** (sul **bus1** a questo punto si trova il valore 1).

Corrige:

Reg_1_to_bus2 <= '1'	Mette in bassa impedenza il **tristate-buffer** connesso tra l'uscita del registro **1** e il **bus2** (sul **bus2** a questo punto si trova il valore 1).

STATO 5, pag. 140: errore di battitura nella spiegazione:

Errata:

Sel_Mux_Read <= '1'	Imposta come uscita del multiplexer connesso all'ingresso del registro **Read** il contenuto del registro **0**.

Corrige:

Sel_Mux_Read <= '1'	Imposta come uscita del multiplexer connesso all'ingresso del registro **Read** il contenuto del registro **1**.

STATO 6, pag. 140: errore di assegnazione del valore al segnale Sel_Mux_Read:

Errata:

Sel_Mux_Read <= '1'	Imposta come uscita del multiplexer connesso all'ingresso del registro **Read** il contenuto del registro **0**.

Corrige:

Sel_Mux_Read <= '0'	Imposta come uscita del multiplexer connesso all'ingresso del registro **Read** il contenuto del registro **1**.

STATO 7, pag. 141: errore di assegnazione del valore di uscita del comparatore:

Errata: "**Caso** comparator_out = '1', ovvero **temp2>temp1**"
Corrige: "**Caso** comparator_out = '0', ovvero **temp2>temp1**"

STATO 7, pag. 142: errore di assegnazione del valore di uscita del comparatore:

Errata: "**Caso** comparator_out = '0', ovvero **temp2<temp1**"
Corrige: "**Caso** comparator_out = '1', ovvero **temp2<temp1**"

Listato VHDL, pagg. 148-167: denominazione Errata del segnale Alu_out_to_bus3:

Errata: il segnale indicato come Reg_out_to_bus3 in tutto il codice VHDL corrisponde al segnale Alu_out_to_bus3.
Corrige: sostituire la denominazione del segnale Reg_out_to_bus3 con Alu_out_to_bus3.

Listato VHDL, pagg. 154: assegnazione Errata del segnale Sel_Mux_Read nello stato 6:

Errata: `Sel_Mux_Read <= '1';`
Corrige: `Sel_Mux_Read <= '0';`

Listato VHDL, pagg. 155: assegnazione Errata del segnale di uscita del comparatore nello stato 7:

Errata: `IF comparator_out = '1' THEN`
Corrige: `IF comparator_out = '0' THEN`

Listato VHDL, pagg. 166: mancata assegnazione del segnale Next_State nello stato OTHERS:

Errata: `Next_State <= 1;` *(manca)*
Corrige: `Next_State <= 1;` *(aggiungere)*

- Esercizio n. 61 - pag. 168: *Manca la seguente*

***NOTA IMPORTANTE**:* Nel processo di assegnazione dei valori di uscita non vengono assegnati i valori a tutti i segnali in ogni stato. Sebbene da un punto di vista funzionale il codice VHDL risulti corretto, è da notare che in fase di sintesi verrà aggiunto un latch per ogni segnale di uscita al fine di mantenerne il valore tra uno stato e il successivo (questo effetto è noto come "latch inferred"). Inoltre, all'uscita dalla condizione di reset la macchina a stati si ritroverà nello stato zero in cui non viene specificato il valore di tutti i segnali. In questo modo il valore della maggior parte dei segnali risulterà indeterminato.

- Esercizio n. 62 - pag. 174: *Idem come Es. 61*

- Esercizio n. 63 - pag. 181: *errore immagine in fondo alla pagina:*

Errata: il segnale MEM_DATA è associato a una freccia con un'unica direzione da RAM verso ASIC
Corrige: il segnale MEM_DATA è di tipo bidirezionale e quindi va raffigurato tramite una freccia bidirezionale

- Esercizio n. 63 - pag. 181: *Idem come Es. 61*

- Esercizio n. 64 - pag. 194: *errore nel diagramma ASM:*

  Errata: Nello stato 13 qualora i sia minore di R-1 la macchina a stati si porta nello stato 4 dove il segnale j viene incrementato di uno dopo che nello stato 12 era stato correttamente impostato a zero. In questo modo per i diverso da zero, j non varrà mai zero.

  Corrige: Una possibile soluzione potrebbe essere aggiungere uno stato 14 in cui la macchina va se la condizione nello stato 13 risulta vera. In questo stato 14 si effettuano gli stessi test dello stato 4 e, a seconda dei risultati dei test, la macchina si porterà dallo stato 14 allo stato 5, 7 o 9.

- Esercizio n. 64 - pag. 195: *valore e tipo errato per le costanti C e R:*

  Errata:
  ```
  constant C : integer := 127;
  constant R : integer := 127;
  ```
  Le costanti C e R sono dichiarati integer e viene loro assegnato il valore 127. Poiché, ad esempio, nel codice VHDL viene effettuata una comparazione tra il segnale i di tipo std_logic_vector e la costante R di tipo integer si lascia al sintetizzatore la scelta di come convertire il valore integer in un std_logic_vector: in linea teorica un sintetizzatore poco "smart" potrebbe usare 1000 bit per rappresentare il numero 127 con un enorme spreco di risorse. Inoltre il valore 127 è errato poiché, ad esempio, la condizione allo stato 11 testa il valore di j con 126 e non 127 come dovrebbe essere poiché si va dal pixel 0 al pixel 127.

  Corrige:
  ```
  constant C : std_logic_vector(14 downto 0)
               := conv_std_logic_vector(128,
               15);
  constant R : std_logic_vector(14 downto 0)
               := conv_std_logic_vector(128,
               15);
  ```

- Esercizio n. 64 - pag. 192: *Manca la seguente*

  ***NOTA IMPORTANTE**:* Gli stati 5, 7 e 9 potrebbero essere eliminati. L'assegnazione effettuata in questi stati potrebbe essere fatta in maniera combinatoria esternamente alla macchina a stati o, alternativamente, in maniera sequenziale tenendo però in questo caso conto del ritardo di un ciclo di clock tra la variazione del segnale in ingresso e in uscita del registro.

- Esercizio n. 64 - pag. 198: *Manca la seguente*

  ***NOTA IMPORTANTE**:* Nello stato 11 viene posto il segnale mem_data2 in alta impedenza ma tale assegnazione risulta superflua poiché il segnale write e read verso la memoria vengono posti entrambi a zero e mem_data2 non è un bus a cui accedono diverse risorse. Seppur non causando un errore funzionale, il sintetizzatore aggiungerà un buffer tri-state non necessario per il corretto funzionamento.

- Esercizio n. 68 - pag. 213: *errore nella dimensione dei segnali in fase di dichiraione:*

  Errata:
  ```
  signal pc: std_logic_vector(23 downto 0);
  signal mar: std_logic_vector(23 downto 0);
  ```
  Corrige:
  ```
  signal pc: std_logic_vector(21 downto 0);
  signal mar: std_logic_vector(21 downto 0);
  ```

- Esercizio n. 68 - pag. 214: *segnali non necessari nella sensitivity list del primo processo:*

  Errata:
  ```
  main_next: PROCESS (state, irq,
  active_mem, active_alu, reset)
  ```
  Corrige:
  ```
  main_next: PROCESS (state, irq)
  ```

- Esercizio n. 68 - pag. 215, codice VHDL, 7ma riga da sotto: *selezione dimensione del segnale ir:*

  Errata:
  ```
  5))), ir(3 downto 0));
  ```
  Corrige:
  ```
  5))), ir(4 downto 0));
  ```

- Esercizio n. 68 - pag. 215, codice VHDL, 10ma riga da sopra: *assegnazione dimensione del segnale pc:*

  Errata:
  ```
  pc<=conv_std_logic_vector(boot_address,
  24);
  ```
  Corrige:
  ```
  pc<=conv_std_logic_vector(boot_address,
  22);
  ```

- Esercizio n. 68 - pag. 215, codice VHDL, 11ma riga da sopra: *assegnazione a '0' dei segnali che attivano le unita' esterne:*

  Errata:
  ```
  when 1 =>
          mar<=pc;
  ```
  Corrige:
  ```
  when 1 =>
          mar<=pc;
          active_mem <= '0';
          active_alu <= '0';
  ```

- Esercizio n. 68 - pag. 215: *Manca la seguente*

  ***NOTA IMPORTANTE**:* Nella condizione di reset non vengono assegnati i valori a molti segnali (*pc, ir, mar, regs, active_alu, active_mem*). In questo modo il firmware potrebbe funzionare correttamente ma sicuramente non si ha il pieno controllo dello stato di partenza dei segnali suddetti. Vale inoltre quanto detto nella NOTA IMPORTANTE dell'esercizio 61.

- Esercizio n. 68 - pag. 215: *Manca la seguente*

NOTA IMPORTANTE: Nella soluzione proposta nel testo i segnali *active_mem, active_alu*, che attivano le unita' funzionali separate, vengono assegnati nel processo che implementa le operazioni sui dati, sensibile al clock. Questo fa si' che tali segnali si attivino sul fronte di clock e che l'operazione della corrispondente unita' funzionale avvenga sul fronte si clock successivo (perdita di un ciclo). Per evitare questo inconveniente, le assegnazioni dei segnali *active_mem, active_alu*, andrebbero piu' opportunamente inserite nel processo che assegna il valore dello stato prossimo, che e' un processo combinatorio e non attende il fronte di clock.

- Esercizio n. 69 - pag. 216, codice VHDL: *mancanza condizione su active_mem per l'assegnazione dello stato prossimo del processo mem_unit_calcola_next:*

Errata:
```
if ir(13)='0' then
   next_state<=2;
else
   next_state<=3;
end if;
```
Corrige:
```
if active_mem = '1' then
    if ir(13)='0' then
         next_state<=2;
    else
         next_state<=3;
    end if;
 else
    next_state<=3;
 end if;
```

- Esercizio n. 69 - pag. 217, codice VHDL: *Errata condizione sul segnale active_mem nel processo mem_unit_assegna_next:*

Errata:
```
ELSIF ck'event and ck='1' and
active_mem='1' then
```

Corrige:
```
ELSIF ck'event and ck='1' then
```

- Esercizio n. 69 - pag. 217, codice VHDL: *mancanza condizione e segnale di reset nella sensitività list:*

Errata:
```
Mem_unit_ops: PROCESS (ck)
    BEGIN
      IF ck'event and ck='1' THEN
```
Corrige:
```
Mem_unit_ops: PROCESS (reset, ck)
    BEGIN
      IF reset = '1' THEN
```

```
            read <= '0';
            write <= '0';
            mar <= (others => '0');
            regs <= (others => '0');
            mem <= (others => 'Z');
        ELSIF ck'event and ck='1' THEN
```

- Esercizio n. 78 - pag. 232: *risultato errato nel calcolo del CPload:*

  Errata: =0.12·4+0.88·1=1.48
  Corrige: =0.12·4+0.88·1=1.36

- Esercizio n. 83 - pag. 236, 9ª riga dal basso: *frase poco chiara:*

  Errata: La DIV che scrive su r8 impone due cicli di clock di distanza con lo ST che legge r8, questa latenza è stata sfruttata inserendovi gli incrementi di r2 e r6 che possono essere eseguiti subito.
  Corrige: L'istruzione DIV che scrive su r8 impone una latenza di due cicli di clock prima che l'istruzione ST ne utilizzi il risultato. I due cicli di latenza sono stati quindi utilizzati per il calcolo dell'indirizzo di b[i] nel registro r3 e l'incremento di r6.

- Esercizio n. 85 - pag. 238, ultimo capoverso dell'esercizio: *valori calcolati in modo errato:*

  Errata: In totale, trascurando sempre le inizializzazioni, si completano 14 operazioni in 21 cicli di clock, dunque un CPI medio pari a 1,57. Le 100 iterazioni richiedono 2100 cicli di clock dei quali 800 spesi a fare delle NOP.
  Corrige: In totale, trascurando sempre le inizializzazioni, si completano 14 operazioni in 21 cicli di clock, dunque un CPI medio pari a 1,5. Le 100 iterazioni richiedono 2100 cicli di clock dei quali 700 spesi a fare delle NOP.

- Esercizio n. 86 - pag. 239, tabella delle latenze: *valore errato della latenza di LD:*

  Errata:

	MUL, DIV	LD	ST, ADD, MOV, CMP, BRGT
Tutte	2	1	0

  Corrige:

	MUL, DIV	LD	ST, ADD, MOV, CMP, BRGT
Tutte	2	2	0

- Esercizio n. 88 - pag. 243: *ultima riga della tabella:*

  Errata: ultima riga della tabella
  Corrige: eliminare ultima riga della tabella

- Esercizio n. 91 - pag. 246, 11[a] riga del primo capoverso: *stringa riportata in maniera Errata:*

  Errata: "[…] infatti l'operazione SHFR R15, R6, #7 restituisce il valore '2' […]"
  Corrige: "[…] infatti l'operazione SHFR R12, R6, #7 restituisce il valore '2' […]"

- Esercizio n. 94 - pag. 257, ultima formula:

  Errata: $n^{\circ}_{cicli} = 2 * 1 + \frac{12 * 128 * 128}{2} + 1 = 180227$

  Corrige: $n^{\circ}_{cicli} = 2 * 1 + \frac{22 * 128 * 128}{2} + 1 = 180227$

- Esercizio n. 96 - pag. 263, 3[a] riga dall'alto: *omesse due parentesi e una negazione:*

  Errata: (C_{in}BA+...+AB'C_{in}' = C_{in}(AB+A'B')+...
  Corrige: (C_{in}BA+...+AB'C_{in}')' = (C_{in}(AB+A'B')+...

- Esercizio n. 102 - pag. 271, 1[a] mappa di Karnaugh: *elemento errato nella mappa:*
  Errata: c'è uno zero nella mappa
  Corrige: tutti gli elementi della mappa valgono 1

- Esercizio n. 102 - pag. 271, 2[a] mappa di Karnaugh: *elemento errato nella mappa:*
  Errata: c'è uno zero nella prima riga della mappa
  Corrige: tutti gli elementi della prima riga della mappa valgono 1

- Esercizio n. 103 - pag. 274, figura in alto: *manca un segnale in ingresso ad un transistor:*
  Errata: segnale mancante nel pull-down nella parte destra della figura
  Corrige: porre il segnale B' ove mancante

- Esercizio n. 107 - pag. 281, figura: *in figura c'è un PMOS in un pull-down:*
  Errata: PMOS nel pull-down del 3° stadio
  Corrige: sostituire il simbolo del PMOS con il simbolo di un NMOS

- Esercizio n. 108 - pag. 284, 2ª riga dall'alto: *errato riferimento a figura:*

  Errata: "La figura sottostante mostra il risultato finale del circuito appena descritto."
  Corrige: "La figura precedente mostra il risultato finale del circuito appena descritto."

- Esercizio n. 108 - pag. 284, 2ª riga dal basso: *errato riferimento a K-map:*

  Errata: "la K-map qui soto."
  Corrige: "la K-map precedente."

- Esercizio n. 108 - pag. 284, 1ª riga dal basso: *espressione logica Errata:*

  Errata: AC+AB=(A+B)C
  Corrige: AC+BC=(A+B)C

- Esercizio n. 108 - pag. 285, 2ª riga dall'alto: *errore nel testo:*

  Errata: "Occupiamoci ora della logica NORA con il primo stadio a precarica bassa e il secondo stadio a precarica bassa. Procediamo […]"
  Corrige: "Occupiamoci ora della logica NORA con il primo stadio a precarica bassa e il secondo stadio a precarica alta. Procediamo […]"

- Esercizio n. 108 - pag. 285: *errato posizionamento della K-map:*

  Errata: la K-map è posizionata dopo la frase che comincia nel seguente modo "Come già visto nel caso di logica Domino […]"
  Corrige: la K-map va posizionata dopo la frase "Sintetizziamo quindi la funzione con la K-map:"

- Esercizio n. 109 - pag. 286: *errato posizionamento del testo:*

  Errata: il testo "Dagli implicanti a 1 avremo f_{PUN1} =(AB)+C. Lo schema circuitale è riportato qui sotto." è in una posizione Errata.
  Corrige: posizionare la frase in oggetto a pagina 285 prima del circuito.

- Esercizio n. 109 - pag. 286, 10ª riga dal basso: *errore nel testo:*

  Errata: "(si veda es. 12)"
  Corrige: eliminare il testo in oggetto.

- Esercizio n. 109 - pag. 289, 8ª riga dal basso: *errore nel testo:*

  Errata: "(vedi prima figura es. 14)"
  Corrige: eliminare il testo in oggetto.

- Esercizio n. 109 - pag. 290, 7ª riga dall'alto: *errore nel testo:*

  Errata: "(si faccia riferimento alla seconda figura es. 14)"
  Corrige: eliminare il testo in oggetto.

- Esercizio n. 109 - pag. 291, 8ª riga dall'alto: *errore nel testo:*

  Errata: "(si faccia riferimento alla terza figura es. 14)"
  Corrige: eliminare il testo in oggetto.

- Esercizio n. 116 - pag. 304: *omessi i segnali sui gate di due transistor:*

  Errata: due PMOS in alto a sinistra nella figura non hanno l'indicazione dei segnali sui gate.
  Corrige: i due PMOS hanno CK sui gate.

- Esercizio n. 118 - pag. 307: *figure errate:*

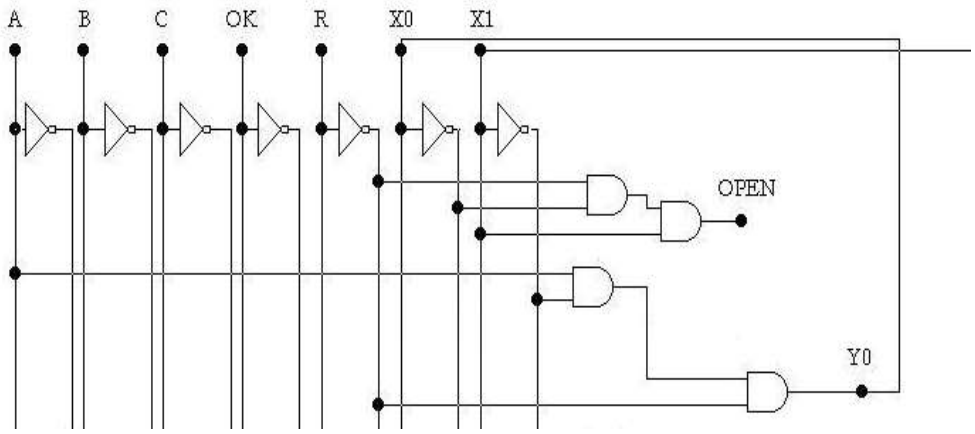
  Errata: Nel diagramma a blocchi e nel diagramma temporale è presente il segnale STR.
  Corrige: Il segnale STR deve essere eliminato da entrambe le figure.

- Esercizio n. 120 - pag. 312, espressione logica di Y_0 : *errore nell'ultimo passaggio:*

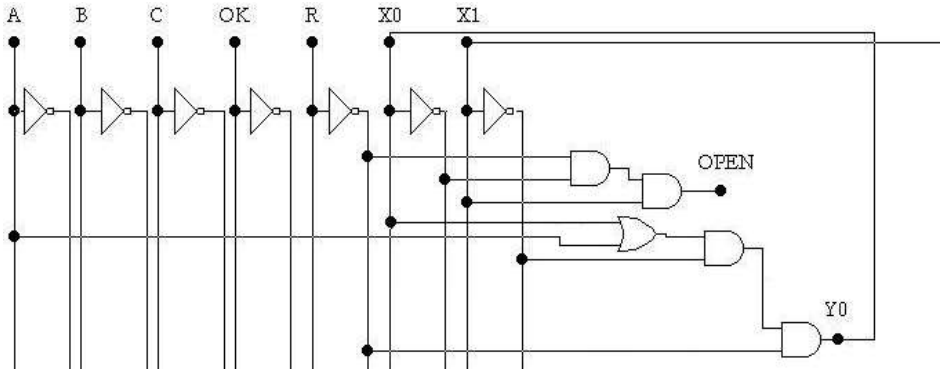
  Errata: = A and (not X_1) and (not R)
  Corrige: = (A or X_0) and (not X_1) and (not R)

- Esercizio n. 120 - pag. 313, circuito logico:

Errata:



Corrige:

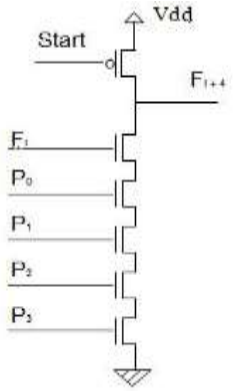


- Esercizio n. 122 - pag. 316, espressione logica di F_{i+4}:

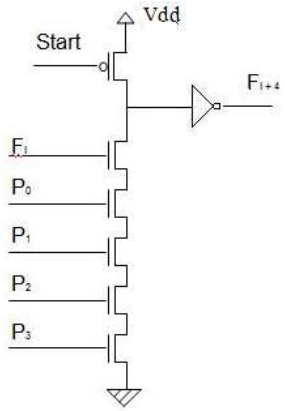
Errata: $F_{i+4}=P_1+P_2+P_3+P_4+F_i$
Corrige: $F_{i+4}=P_1P_2P_3P_4F_i$

- Esercizio n. 122 - pag. 317, figura in basso: *manca un inverter sull'uscita:*

Errata:



Corrige:



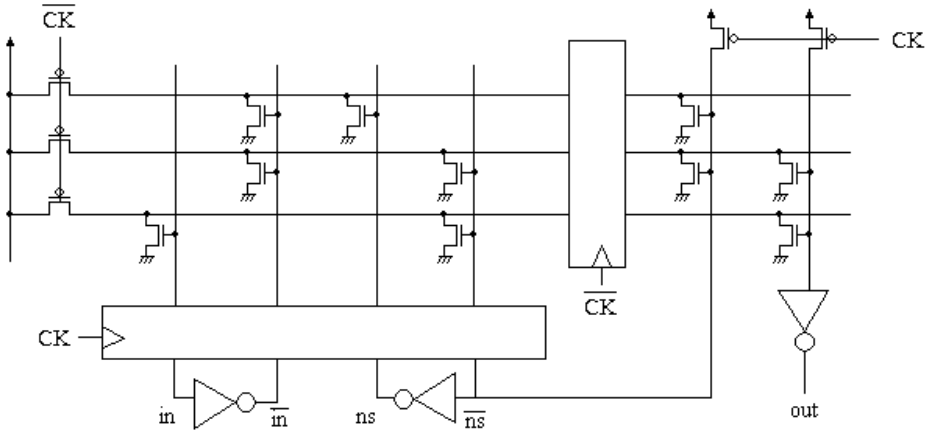
- Esercizio n. 122 - pag. 317, 9ª riga dal basso:

  Errata: "Quando Start viene posto a valore logico basso per la fase di precarica, il circuito inizializza F_{i+4} a '1'. Quando Start è portato a livello alto il circuito entra in fase di valutazione e quindi l'uscita F_{i+4} va a '0' quando tutti i propagate ed F_i sono a livello alto."

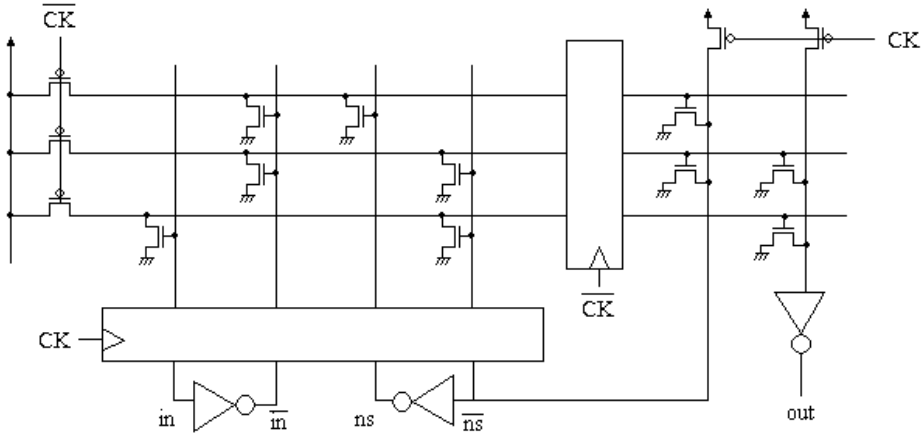
  Corrige: "Quando Start viene posto a valore logico basso per la fase di precarica, il circuito inizializza F_{i+4} a '0'. Quando Start è portato a livello alto il circuito entra in fase di valutazione e quindi l'uscita F_{i+4} va a '1' quando tutti i propagate ed F_i sono a livello alto."

- Esercizio n. 124 - pag. 321, figura PLA esercizio: *connessione transistor nel piano degli OR:*

  Errata:

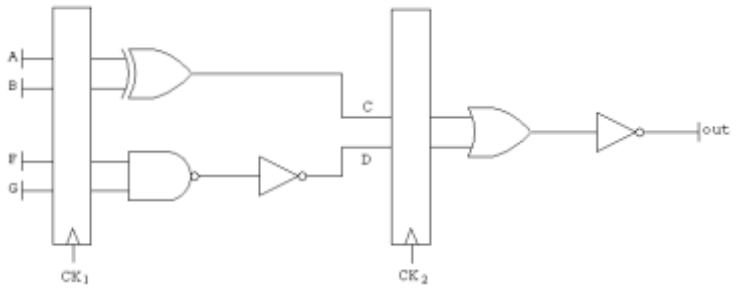


Corrige:

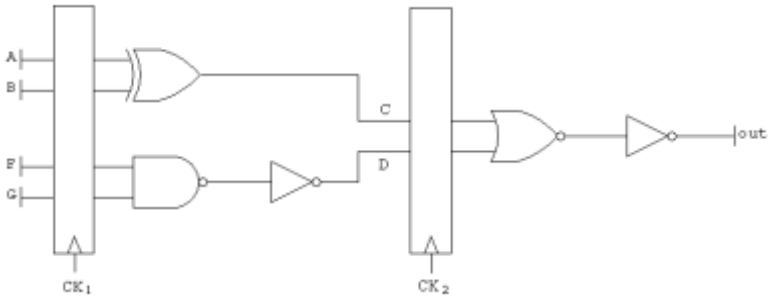


- Esercizio n. 125 - pag. 321, circuito logico dell'esercizio: *porta OR da sostituire con NOR:*

Errata:



Corrige:



- Esercizio n. 129 - pag. 332, titolo esercizio:

Errata: "[…] il cui ritardo sia approssimabile come Tchain = 0.02 M ns."

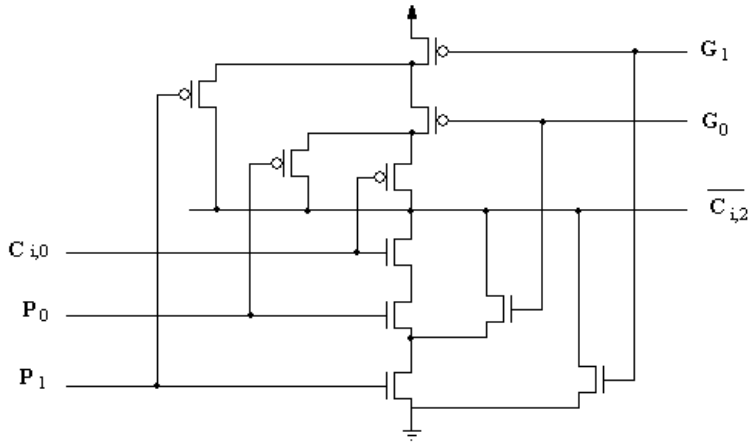
Corrige: …**"[**…] il cui ritardo sia approssimabile come Tchain = 0.02 M^2 ns."

- Esercizio n. 133 - pag. 343, 3ª figura: *è stato omesso il segno di negazione su $C_{i,2}$:*

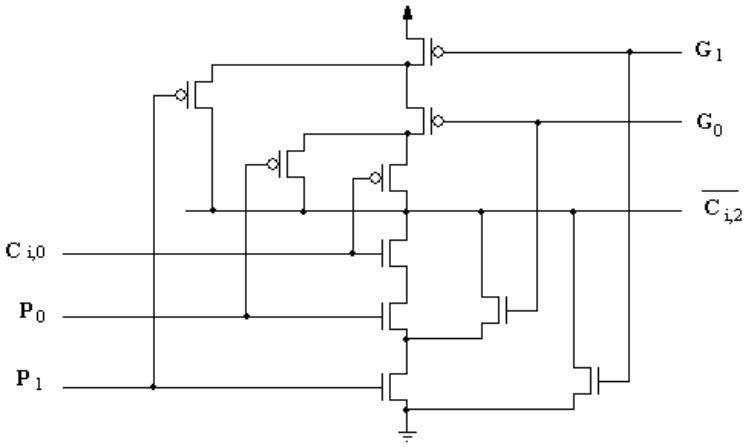
  Errata: L'uscita del primo CLA , $C_{i,1}$ è generata in forma diretta.
  Corrige: L'uscita del primo CLA , $C_{i,1}$, è generata in forma negata.

- Esercizio n. 133 - pag. 344, 1ª figura: *è stato omesso un pallino di collegamento sull'ultimo transistor del pull-down:*

Errata:

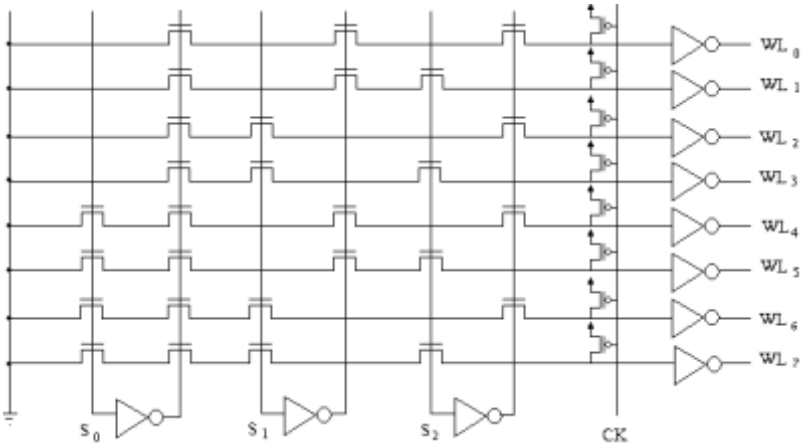


Corrige:

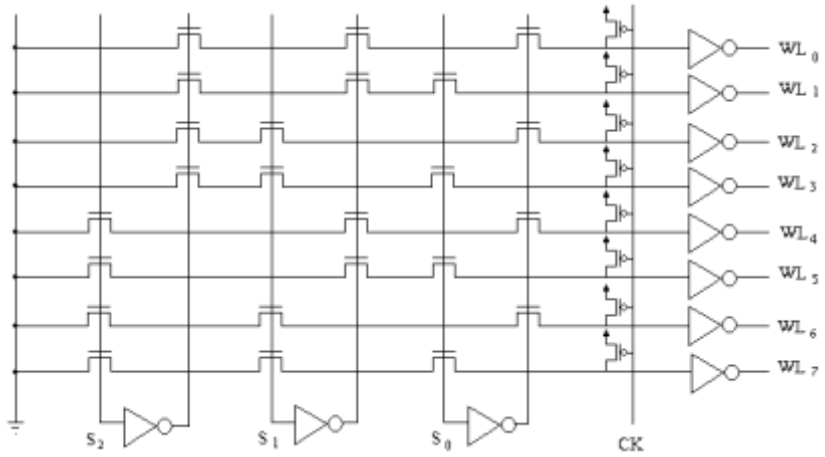


- Esercizio n. 135 - pag. 348, figura in basso (decoder): *posizione S0, S1, S2 e pull down di WL4, WL5, WL6, WL7:*

Errata:



Corrige:



- Esercizio n. 136 - pag. 349, Figura del flip flop statico:

Errata: il pull down delle due catene di inverter sono PMOS. Inoltre i transistor in basso dei due transmission gate pilotati da CK negato (nel primo latch) e dal CK diretto (nel secondo latch) sono NMOS.

Corrige: il pull down delle due catene di inverter devono essere NMOS. I transistor in basso dei due transmission gate pilotati da CK negato (nel primo latch) e dal CK diretto (nel secondo latch) sono NMOS.

- Esercizio n. 140 - pag. 358, Titolo esercizio:

Errata: "[…] utilizzando solo celle CMOS statiche convenzionali a due ingressi [...]"

Corrige: "[…] utilizzando solo celle CMOS statiche convenzionali […]"

- Esercizio n. 141 - pag. 361, Titolo esercizio:

  Errata: "[…] utilizzando solo celle CMOS statiche convenzionali a due ingressi [...]"
  Corrige: "[…] utilizzando solo celle CMOS statiche convenzionali […]"

- Esercizio n. 142 - pag. 364, 5[a] riga dal basso: *calcolo del Path Logical Effort:*

  Errata: $G = \Pi g_i = 6/3 \times 1 \times 5/3 \times 1 = 3.3$
  Corrige: $G = \Pi g_i = 6/3 \times 1 \times 4/3 \times 1 = 2.66$

- Esercizio n. 142 - pag. 364, 3[a] riga dal basso: *calcolo del Path Electrical Effort:*

  Errata: $H = C_{out}/C_{in} = 1$
  Corrige: $H = C_{out}/C_{in} = 2$

- Esercizio n. 142 - pag. 364, 1[a] riga dal basso: *calcolo del Path Effort:*

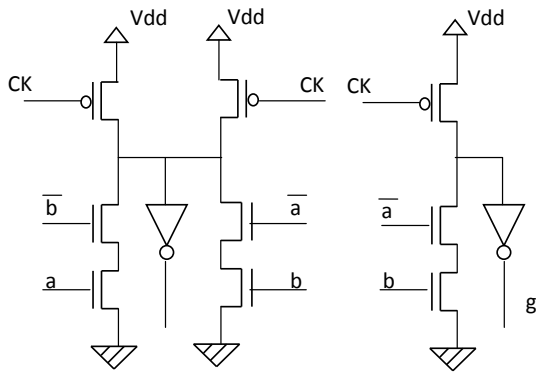
  Errata: $F = GBH = 6.67$
  Corrige: $F = GBH = 5.32$

- Esercizio n. 142 - pag. 365, 1[a] rigo dall'alto:

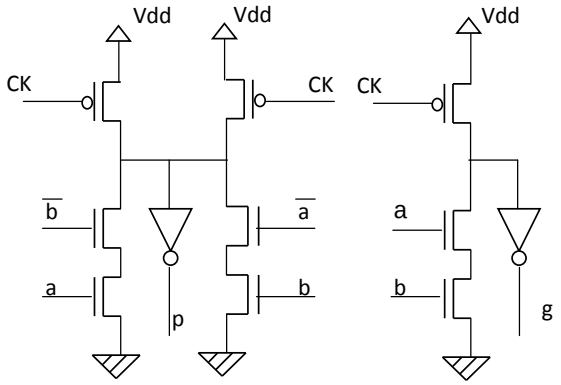
  Errata: "Il ritardo minimo così ottenuto è $D_{min} = N(GBH)^{1/N} + P =$ 14,43 unità […]"
  Corrige: "Il ritardo minimo così ottenuto è $D_{min} = N(GBH)^{1/N} + P =$ 14,07 unità […]"

- Esercizio n. 142 - pag. 365, figura in basso:

  Errata:



Corrige:



- Esercizio n. 145 - pag. 369, 11[a] riga dall'alto:

  Errata: "Da cui si ricava $\alpha_D < 0.142$"
  Corrige: "Da cui si ricava $\alpha_D < 0.2$."

- Esercizio n. 146 - pag. 369, 11[a] riga dal basso:

  Errata: "[…] (in complemento a 2, ultimo bit del risultato pari a '0') [...]"
  Corrige: "[…] (in complemento a 2, il bit più significativo del risultato è pari a '0') [...]"

- Esercizio n. 155 - pag. 384, 11[a] riga dal basso: *il branching effort è pari a due poiché il segnale A in ingresso si dirama su due transistor:*

  Errata: G=4/9; H=1; P=16/3; B=1;
  Corrige: G=4/9; H=1; P=16/3; B=2;

- Esercizio n. 155 - pag. 384, 9[a] riga dal basso: *il calcolo del D_{ABS} cambia in relazione alla precedente correzione:*

  Errata: $D_{ABS} = D_{min} * \tau = N(GBH)^{1/N} + P = 86$ ps
  Corrige: $D_{ABS} = D_{min} * \tau = N(GBH)^{1/N} + P = 9.2\ \tau$

- Esercizio n. 155 - pag. 385, 2[a] riga dall'alto:

  Errata: $D_{ABS} = D_{min} * \tau = N(GBH)^{1/N} + P = 66$ ps
  Corrige: $D_{ABS} = D_{min} * \tau = N(GBH)^{1/N} + P = 6.6\ \tau$