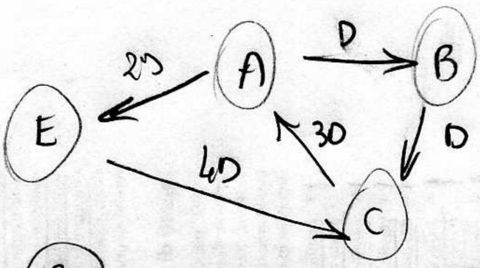
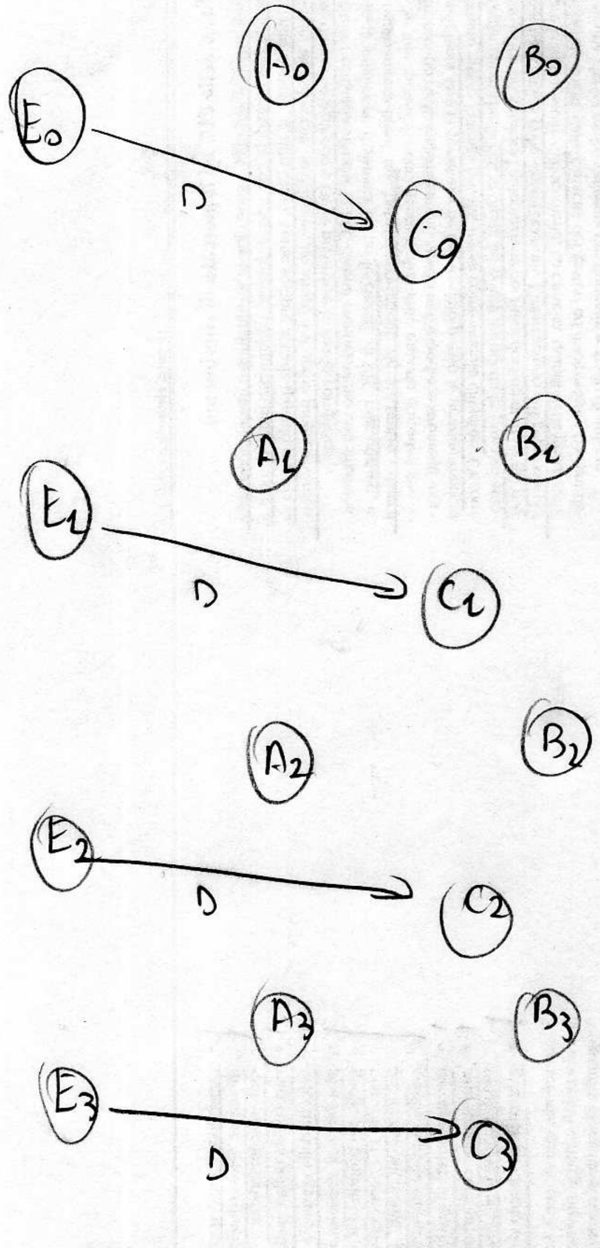
Sistemi Programmabili per telecomunicazioni - Temi di esame (E1)

23/01/2006

2) Dato:



J=4



$A_0 \rightarrow E_j$: $\begin{cases} j = 2\%4 = 2 \\ w_j = 0 \end{cases}$

$A_1 \rightarrow E_j$ $\begin{cases} j = 3\%4 = 3 \\ w_j = 0 \end{cases}$

$A_2 \rightarrow E_j$ $\begin{cases} j = 0 \\ w_j = 1 \end{cases}$

$A_3 \rightarrow E_j$ $\begin{cases} j = 1 \\ w_j = 1 \end{cases}$

$A_0 \rightarrow B_j$ $\begin{cases} j = 1 \\ w_j = 0 \end{cases}$ $A_1 \rightarrow B_j$ $\begin{cases} j = 2 \\ w_j = 0 \end{cases}$

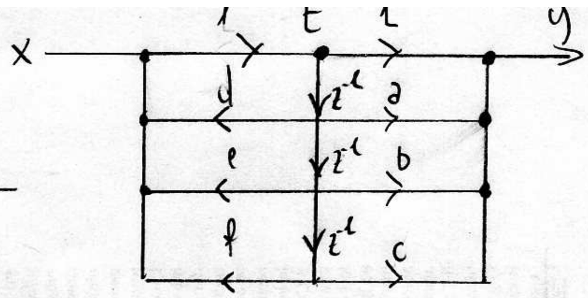
$A_2 \rightarrow B_j$ $\begin{cases} j = 3 \\ w_j = 0 \end{cases}$ $A_3 \rightarrow B_j$ $\begin{cases} j = 0 \\ w_j = 1 \end{cases}$

Usiamo l'astuzia: fatto uno, fatti tutti.
posso dire di getto che:

$$\boxed{\begin{array}{ll} B_0 \rightarrow C_1 & B_1 \rightarrow C_2 \\ w_j = 0 & w_j = 0 \\ B_2 \rightarrow C_3 & B_3 \rightarrow C_0 \\ w_j = 0 & w_j = 1 \end{array}}$$

$C_0 \rightarrow A_j$ $\begin{cases} j = 3 \\ w_j = 0 \end{cases}$ $C_1 \rightarrow A_j$ $\begin{cases} j = 0 \\ w_j = 1 \end{cases}$; $C_2 \rightarrow A_j$ $\begin{cases} j = 1 \\ w_j = 1 \end{cases}$ $C_3 \rightarrow A_j$ $\begin{cases} j = 2 \\ w_j = 1 \end{cases}$

3) Si ha qualcosa tipo ciò:



$$H(z) = \frac{1 + a z^{-1} + b z^{-2} + c z^{-3}}{1 + d z^{-1} + e z^{-2} + f z^{-3}}$$

(E2)

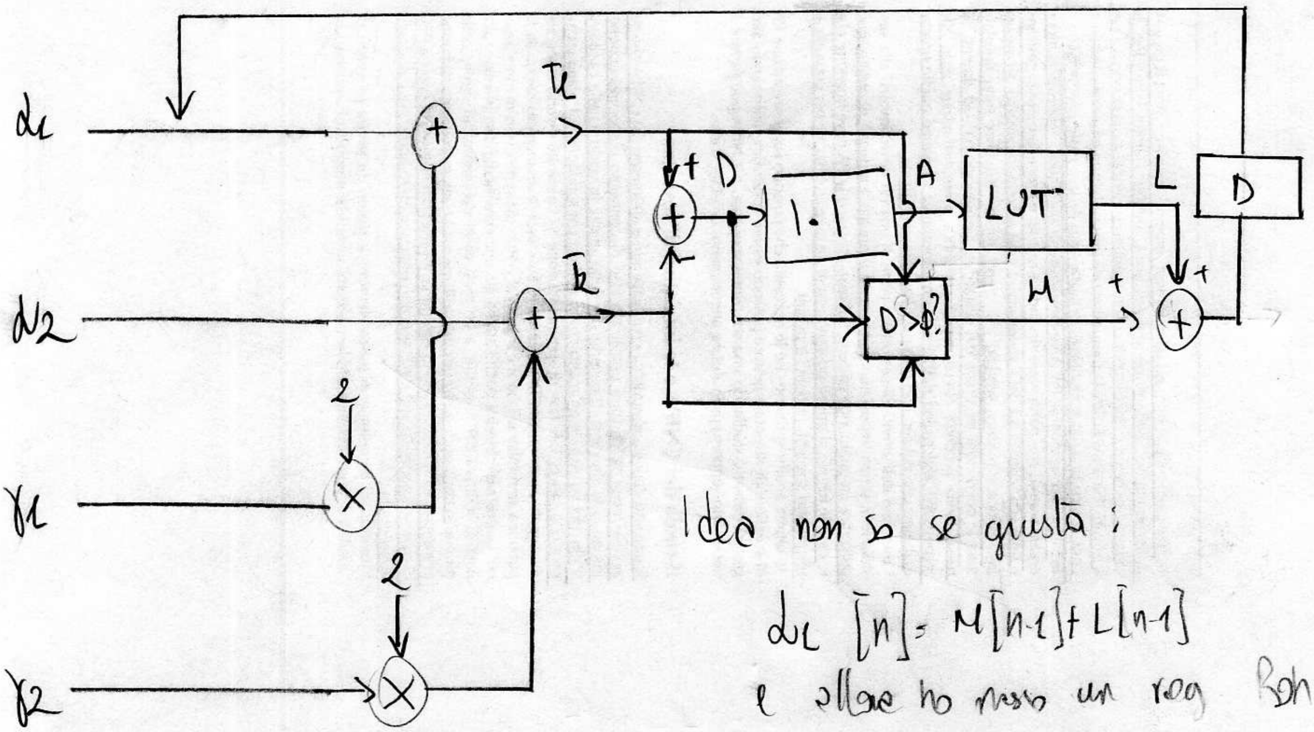
```
ARCHITECTURE Behavioural OF IIR IS
SIGNAL r1, r2, r3, t : std_logic_vector (K-1 downto 0);
-- chiamo "t" il "nodo" centrale.

Begin
Regist : PROCESS(CLK, RSTn)
 BEGIN
 IF RSTn = '0' THEN  r1 <= (OTHERS => '0'); r2 <= (OTHERS => '0'); r3 <= (OTHERS => '0');
 ELSIF (CLK'EVENT AND CLK = '1') THEN
 r1 <= t; r2 <= r1; r3 <= r2;
 END IF;
 END PROCESS;
Comb : PROCESS (x, r1, r2, r3, t); -- non metto i coefficienti...
 BEGIN
 y <= t + a*r1 + b*r2 + c*r3 ;
 t <= x + d*r1 + e*r2 + f*r3 ;
 END PROCESS;
END Architecture;
```

06/02/06 - Versione A

(E3)

1) $T_{add} = 2$; $T_{abs} = 1$; $T_{LUT} = 1$; $T_{IF} = 1$.



Idea non so se giusta:

$\alpha_1[n] = M[n-1] + L[n-1]$

e allora ho messo un reg. Boh.

$T_{cp} = T_{add} + T_{add} + T_{abs} + T_{LUT} + T_{add} = 8. = T_\infty.$

Ho $T_\infty = T_{cp}$: con le tecniche universali, non si può far di meglio.

Posso applicare il Lookahead; su $y[n] = \alpha_1[n+1]$

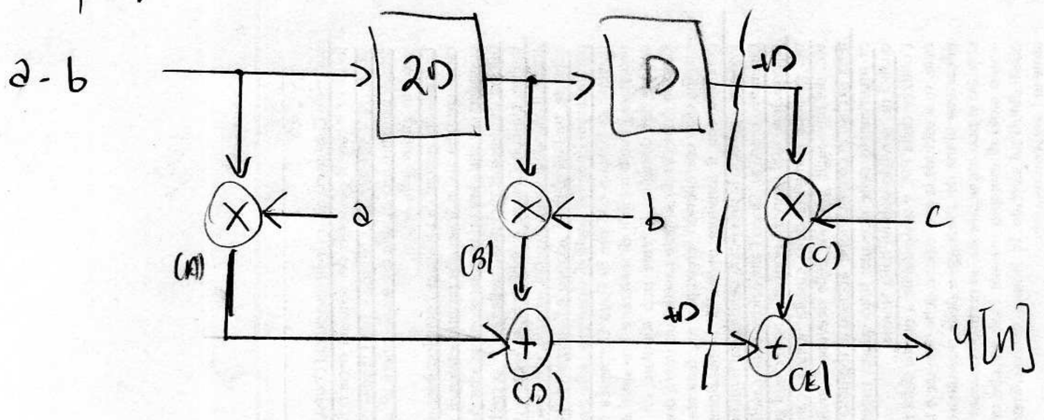
$\alpha_1[n+2] = M[n+1] + L[n+1] = LUT\{|\alpha_1[n+1] + 2\gamma_1[n+1] - \alpha_2[n+1] - 2\gamma_2[n+1]|\} +$
$+ IF\{[\alpha_1[n+1] + 2\gamma_1[n+1] - \alpha_2[n+1] - 2\gamma_2[n+1]], [\alpha_1[n+1] + 2\gamma_1[n+1]], [\alpha_2[n+1] + 2\gamma_2[n+1]]\}$

$\hookrightarrow \alpha_1[n+2] = LUT \ldots$

17/02/06 - Versione A

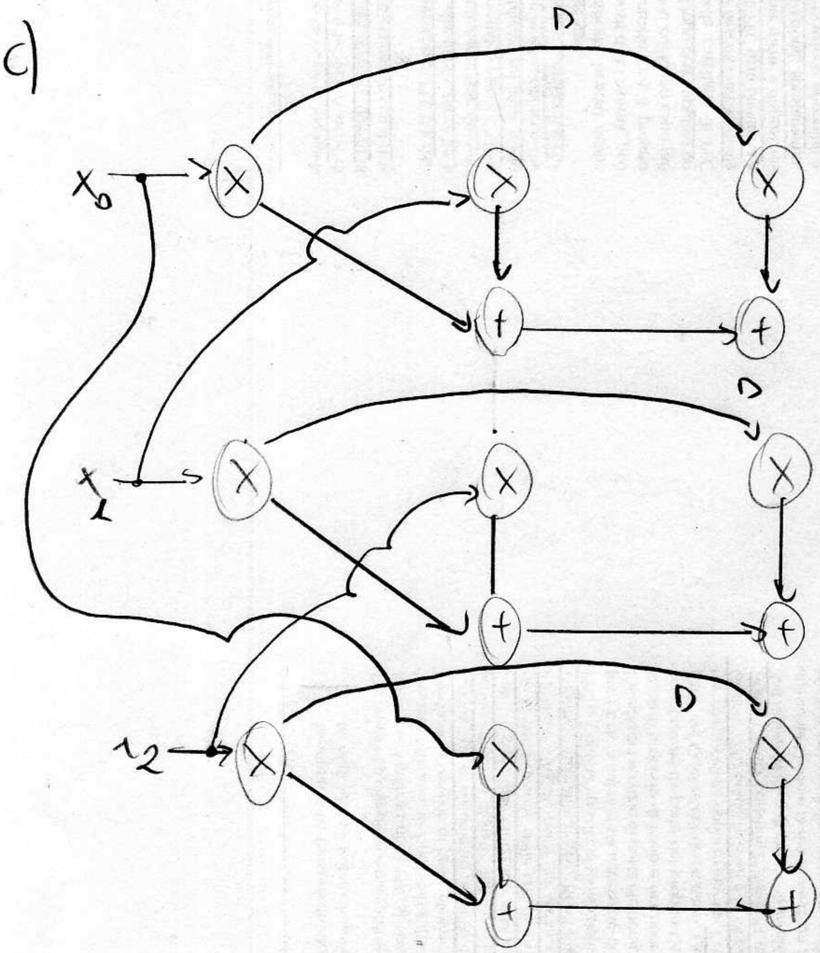
(E5)

1) $y[n] = a\,x[n] + b\,x[n-2] + c\,x[n-3]$



$J = 3$

c)



$x_0 \to B_1 \quad \begin{cases} j = 2 \\ \forall j = 0 \end{cases}$

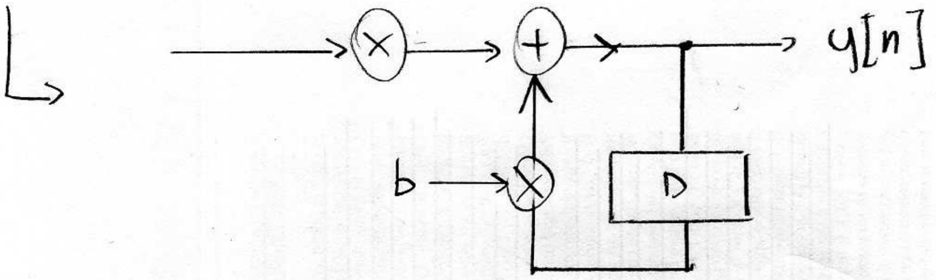
$x_1 \to B_1 \quad \begin{cases} j = 0 \\ \forall j = 1 \end{cases}$

$x_2 \to B_1 \quad \begin{cases} j = 1 \\ \forall j = 0 \end{cases}$

$\boxed{T_{cp}' = T_m + 2T_a}$

$T_s' = 3(T_m + 2T_a) = \frac{1}{3} T_s$

3) $y = a\,x[n] + b\,y[n-1]$ (IIR)

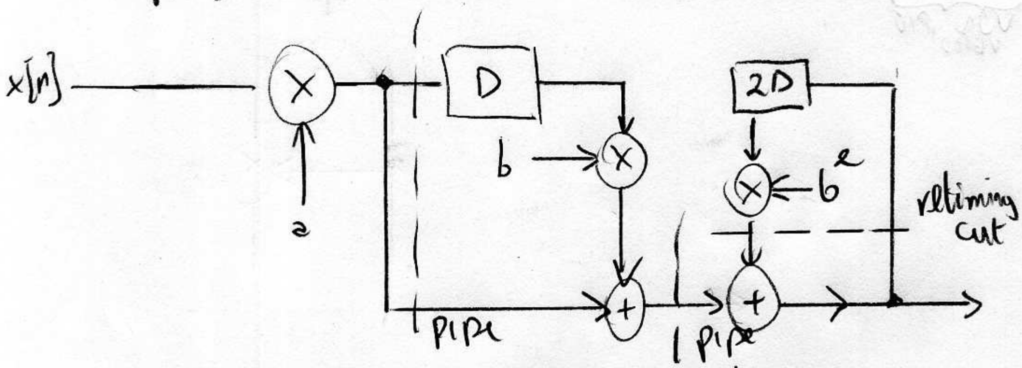


$T_S = T_{cp} = T_{ck} = T_m + T_a = 3.$

Applico il lookahead;

$y[n+1] = a\,x[n+1] + b\,y[n] \longrightarrow y[n+1] = a\,x[n+1] + b\left[a\,x[n] + b\,y[n-1]\right]$

$\longrightarrow y[n] = a\,x[n] + a\,b\,x[n-1] + b^2\,y[n-2].$



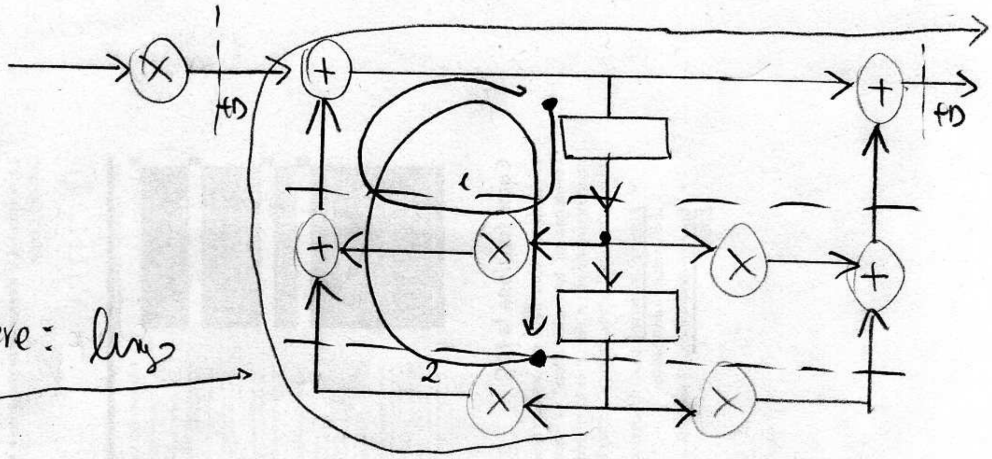
$T_{ck} = T_S = T_{cp} = T_m + 2T_a$ peggiorato!

Ma ciò è ottimizzabile con pipelining e retiming.

03/02/07 (E7)

1) $T_m = 2;\ T_a = 1$

Copio metà del circuito per ottimizzarlo.

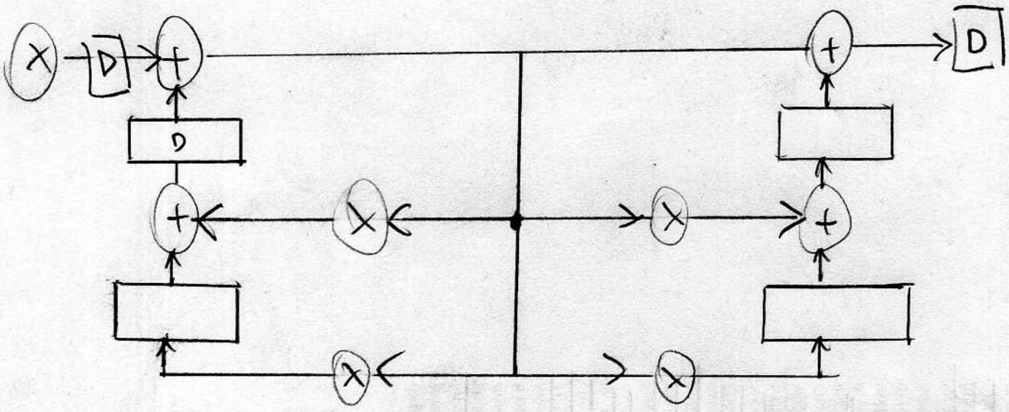


Il ritardo T_{cp} potrebbe essere: lungo questo path:

$T_{cp} = T_m + 5T_a = 7.$

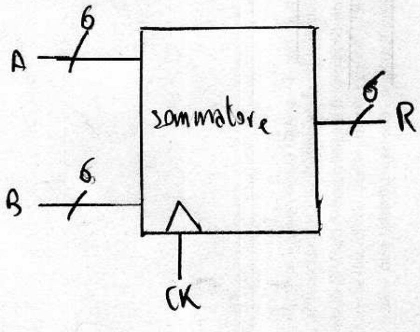
Per quanto riguarda T_∞: ci sono 2 loop. $T_{lp1} = T_m + 2T_a$, $T_{lp2} = \frac{1}{2}(T_m + 2T_a)$ $\longrightarrow$ $\boxed{T_\infty = T_m + 2T_a}$

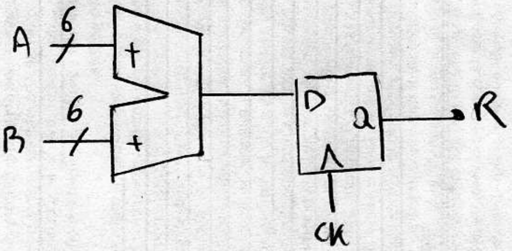
Ciò suggerisce che possiamo fare di meglio. Di sicuro una buona idea è quella di fare dei cut su ingresso e soprattutto uscita. Altro cut, NON feed forward, è quello che collega i registri ai due filtri. Proviamo così:



Idem, per l'altra semi-struttura.

2) Provo a disegnare "sommatore":





1) - con decimazione di <u>tempo</u>. (DIT)

La FFT ha un'eq. di partenza del tipo:

$$X(f) = \sum_{i=0}^{N-1} x[n] W_N^{kn} \quad \text{dove } W_N = \exp\left(-j 2\pi \frac{1}{N}\right)$$

se uso un radix-2, ho:

$$X(f) = \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_N^{2kr} + \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_N^{(2r+1)k} = \sum_{r=0}^{\frac{N}{2}-1} x[2r] W_N^{2kr} + W_N^{K} \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_N^{2rk}$$

Ossia da una da 8, ne ho 2 da 4. Reitero:

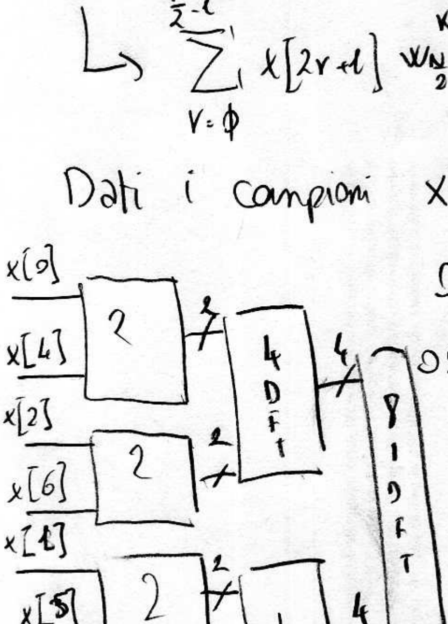
$$\sum_{r=0}^{\frac{N}{2}-1} x[2r] W_{\frac{N}{2}}^{kr} = \sum_{i=0}^{\frac{N}{4}-1} x[2(2i)] W_{\frac{N}{2}}^{2ki} + \sum_{i=0}^{\frac{N}{4}-1} x[2(2i+1)] W_{\frac{N}{2}}^{2k(i+1)} =$$

$$= \sum_{i=0}^{\frac{N}{4}-1} x[4i] W_{\frac{N}{4}}^{ki} + W_{\frac{N}{2}} \sum x[4i+2] W_{\frac{N}{4}}^{ki}$$

Stessa cosa andrebbe fatta per il termine in $2r+1$, sostituendo a r $2i$ e $2i+1$:

$$\hookrightarrow \sum_{r=0}^{\frac{N}{2}-1} x[2r+1] W_{\frac{N}{2}}^{kr} = \sum_{i=0}^{\frac{N}{4}-1} x[4i+1] W_{\frac{N}{2}}^{2kr} + \sum_{i=0}^{\frac{N}{4}-1} x[4i+3] W_{\frac{N}{2}}^{2ki+1}$$

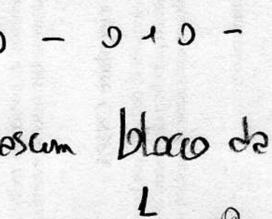
Dati i campioni $x[0 \div N-1]$, ossia 0, 1, 2, 3, 4, 5, 6, 7, prenderò:

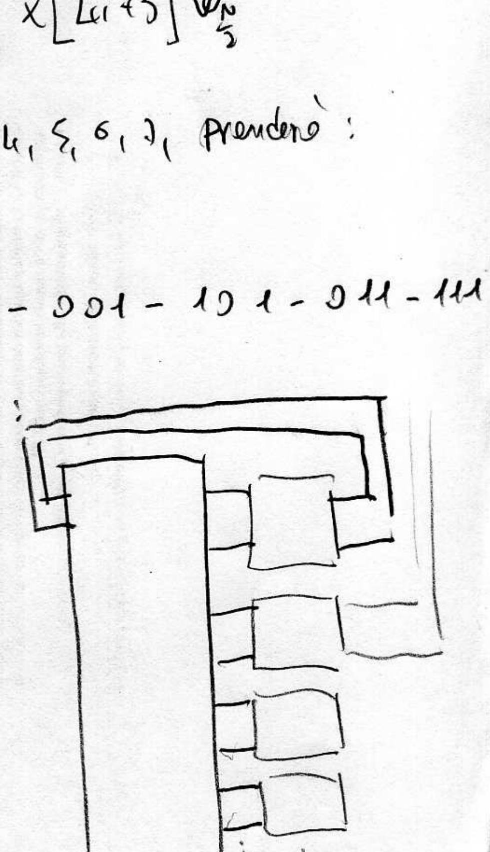


0, 4, 2, 6, 1, 5, 3, 7:

000 - 100 - 010 - 110 - 001 - 101 - 011 - 111

dove ciascun blocco da 2 è:





Un'arch. completamente parallela richiede N log N (N numero di campioni) blocchi da 2-DFT, con una memoria; ciò fa, in 1 ck, elaborare tutto.

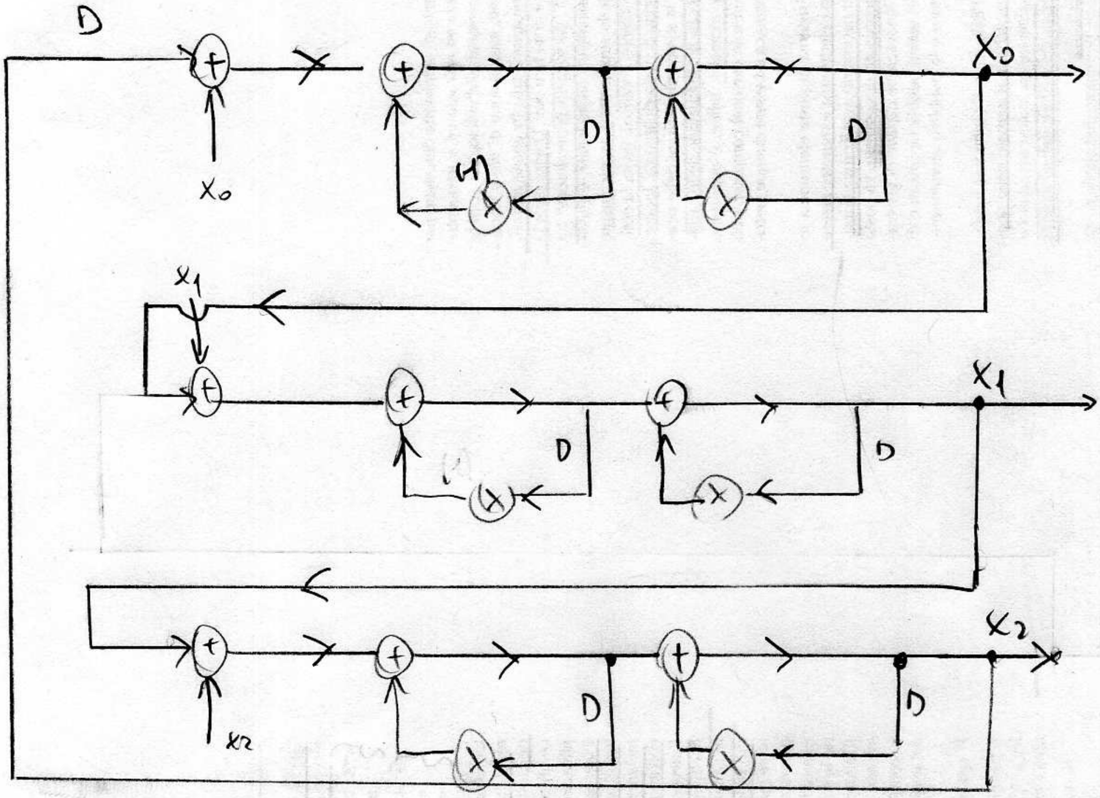
2) Vado per ispezione: ho 2 "gruppi" di cicli: quello 1 (interno), e quello esterno. (E9)

T_{cp}: $T_m + 2T_a$. Parto dal T_m nel ciclo interno sinistro, e vado all'uscita.

$$T_{CK} = T_{cp} = T_S = T_m + 2T_a \longrightarrow f_S = \frac{1}{T_m + 2T_a} = \frac{1}{2+5} = 0{,}1429$$

$$T_\infty = \max\left\{\frac{T_m + T_a}{3}, \frac{3T_a}{1}\right\} = \max\left\{\frac{6}{3}, 3\right\} = 3.$$

Parallelizzo la struttura: $J = 3$; gli unici rami da collegare sono quelli dell'anello esterno:



$$x_0 \longrightarrow A_j \begin{cases} j = 1 \\ w_j = 0 \end{cases} \quad x_1 \longrightarrow A_j \begin{cases} j = 2 \\ w_j = 0 \end{cases} \quad x_2 \longrightarrow A_j \begin{cases} j = 0 \\ w_j = 1 \end{cases}$$

Ricalcolo le grandezze:

T_{cp} = parto da (1) e vado all'uscita x_2, passando per x_0 e x_1:

$\hookrightarrow T_m + T_a + T_a + 3T_a + 3T_a = 13$

Ma:

$T_S' = \frac{1}{3} \times T_{CK}' \Rightarrow f_S = \frac{3}{T_m + 8T_a} = 0{,}231$

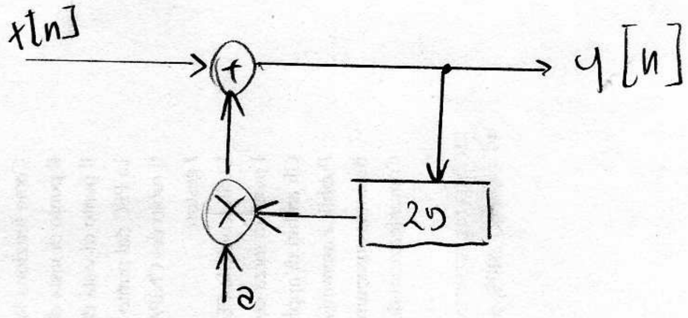
(contro 0,33)

Questa è ancora migliorabile con il pipelining/retiming.

04/09/2008 (E10)

1) Rispondo a parte

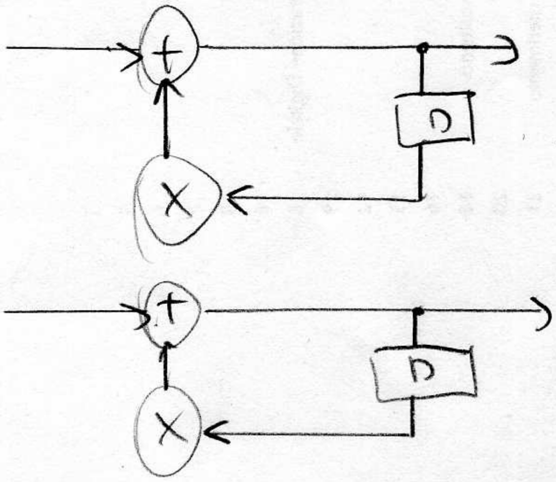
2) $y[n] = x[n] + a\,y[n-2]$;



$T_\infty = \frac{T_m + T_a}{2}$.

$T_{cp} = T_m + T_a$; $T_s = T_{cp} = T_{ck}$

Parallelismo:



$T_s = 2\,T_{ck}' = T_m + T_a$

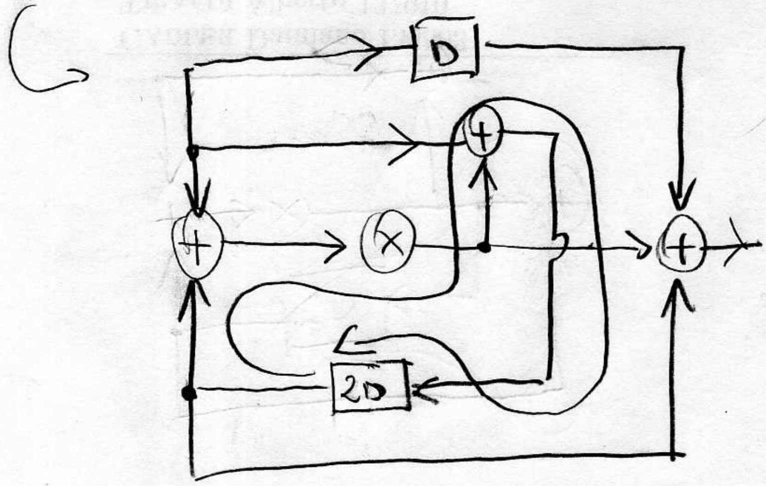
$Th = \frac{2}{T_m + T_a}$

No: facendo Lookahead all'inizio potrei, ma facendo applicazione di una delle tecniche universali, no: folding, unfolding, pipelining, decomposition (resource sharing).

Sistemi Programmabili per Telecomunicazioni (T1)

Esercizi svolti

1) $T_m = 18\,ns$; $T_a = 9\,ns$; osservo la presenza della simmetria (2 cascate di blocchi uguali) e ragiono su 1;



Questo sembra essere l'unico loop. 2 registri, dunque:

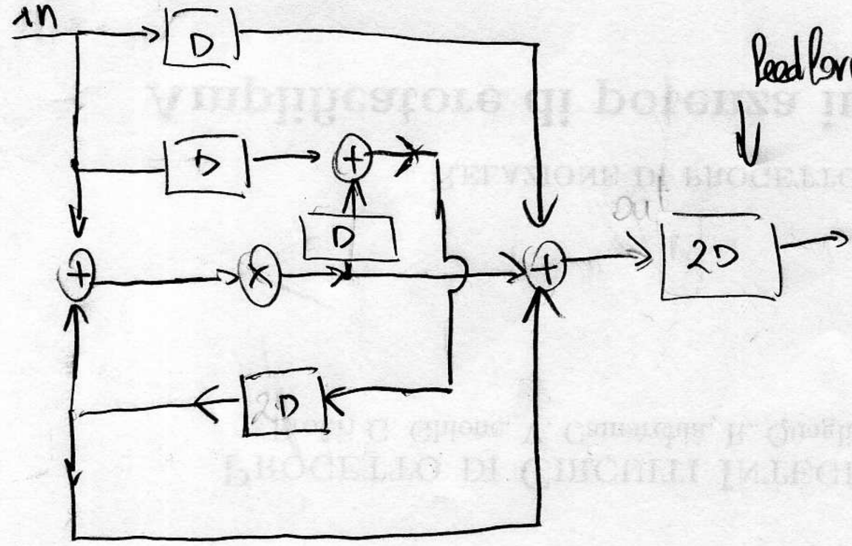
a) $T_\infty = \dfrac{T_m + 2T_a}{2} = 18\,ns$;

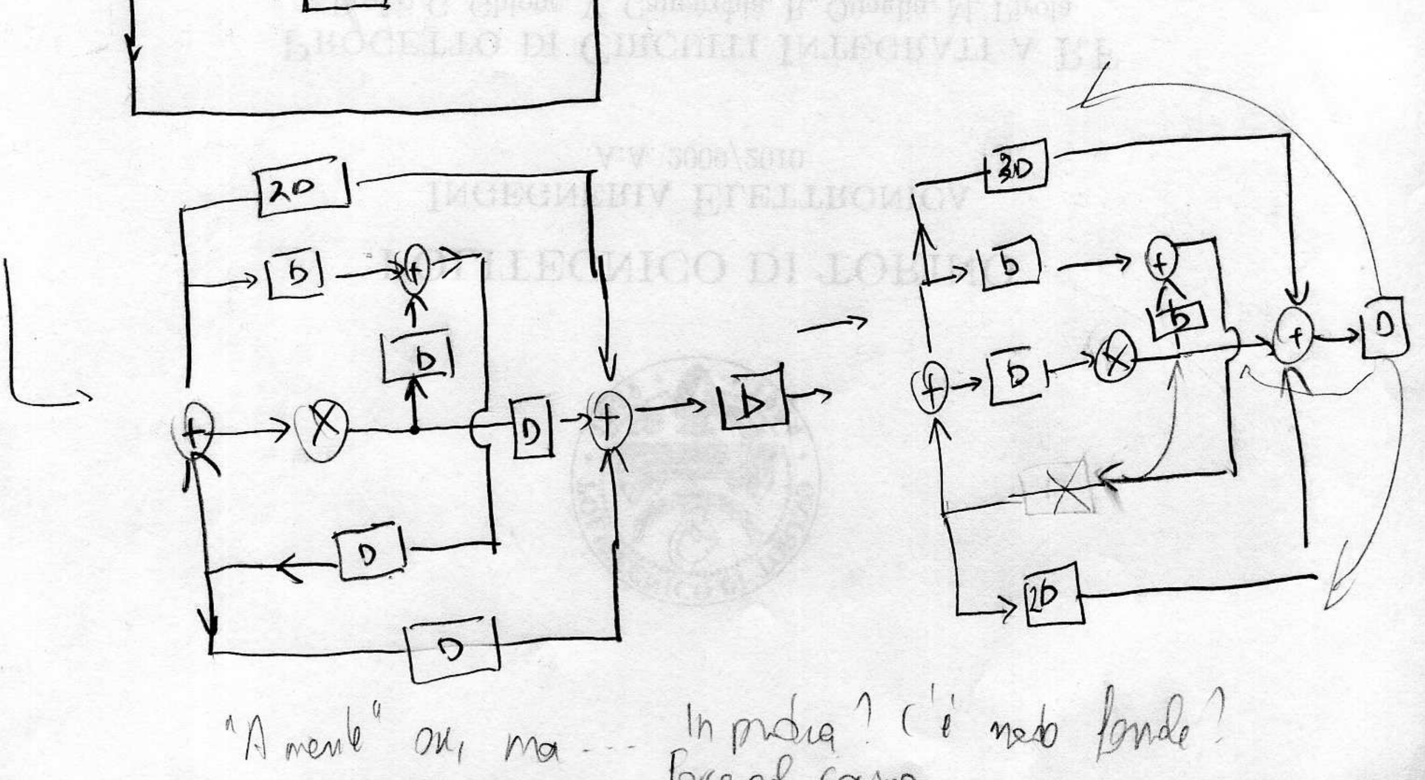
b) $T_{cp} = 88\,ns$

per ispezione

ALTRI MODI?

c) Feedforw. cut set; posso metter registri

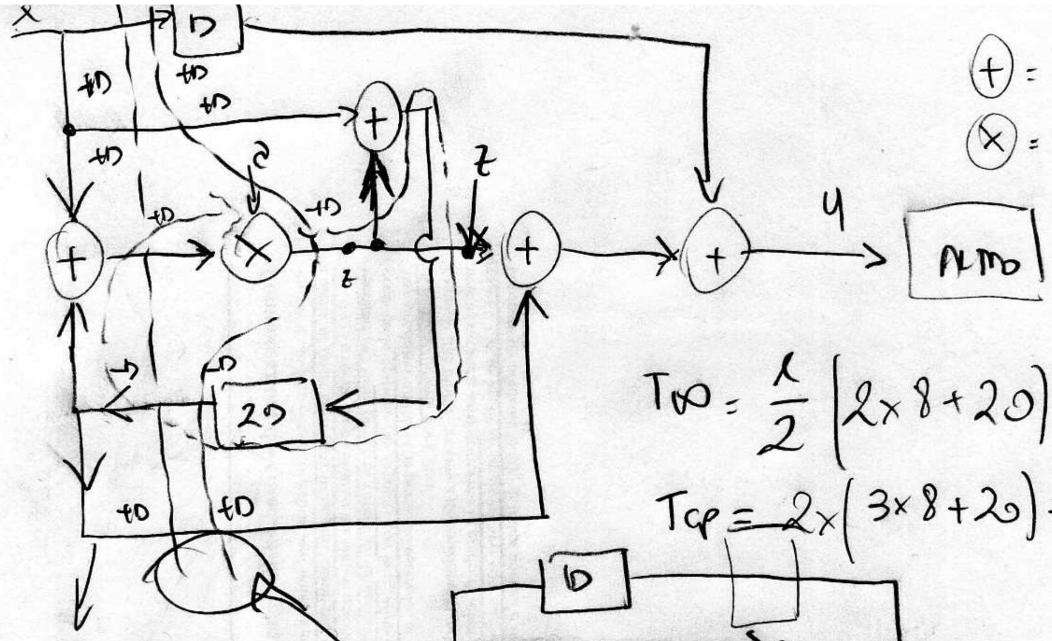




"A mente" ok, ma ... In pratica? C'è modo fonde? Forse col casino....

TL bis

2)



$\oplus$ = 8 ns

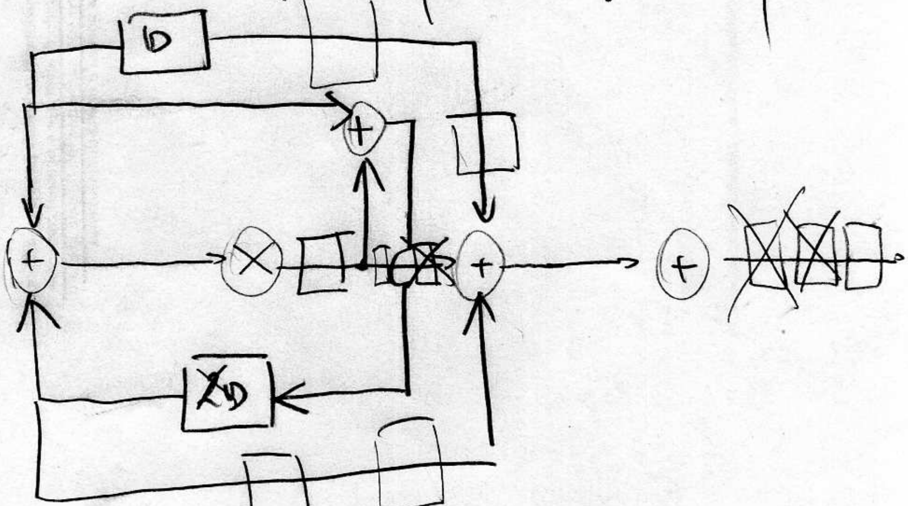
$\otimes$ = 20 ns

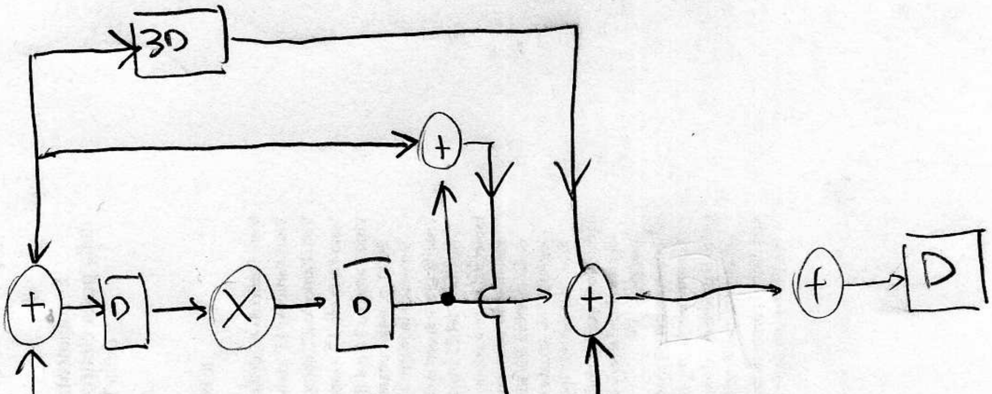
$$T_\infty = \frac{1}{2}\left(2\times 8+20\right) = 18 \text{ ns};$$

$$T_{cp} = 2\times\left(3\times 8+20\right) = 88 \text{ ns};$$

$z[n] = (z[n-2]+x[n])*a$

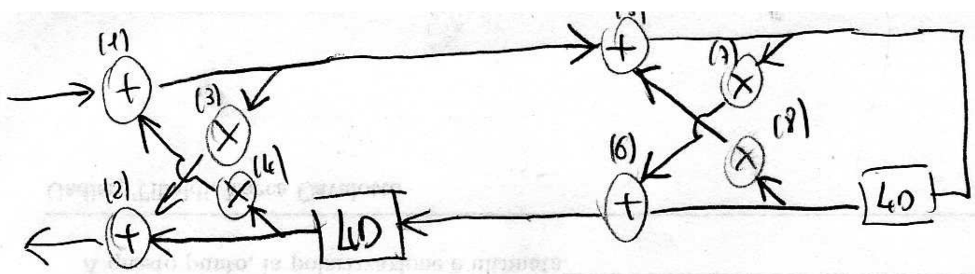
Come puoi farlo: cut set retiming, e scegliere questi tagli





Osservazione: negli esercizi di retiming, usa i cut-set: aggiungi a tutti i rami da un sottografo 1 al 2, e togli lo stesso numero in quelli diretti da 2 a 1.
Cerca il "punto critico" (in questo caso, gli archi che van nel moltiplicatore) e cerca semplicemente di far passar da lì i cuts.

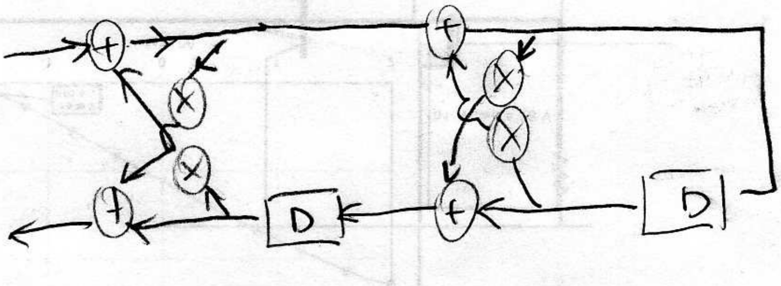
2)



Mia idea: fare unfolding. Se faccio un unfolding di ordine 4, ho:

- $i = j$
- $w_j = \frac{w_i}{J} = D$.

e niente "collegamenti misti". $\Longrightarrow$



Esercizi di autovalutazione

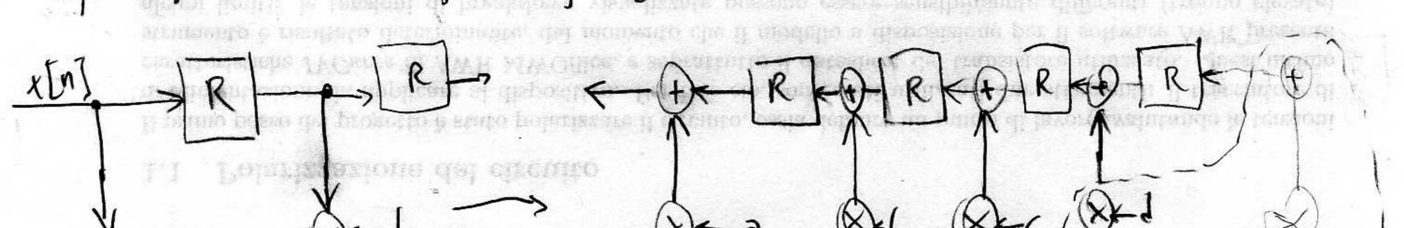
1) a) Si cerchino potenze di 2, sia nella parte intera, sia nella parte frazionaria, facendo attenzione; per esempio, $0{,}625 = 2^{-1}+2^{-3}$; $1{,}25 = 1+2^{-2}$. Si presti attenzione a ciò. In questo caso 0,625 richiede 3 bit per la parte frazionaria, 5 richiede 3 bit per la parte intera; aggiungo 1 bit di segno: 7 bit in tutto.

b) $b_{sum} = \sum |h[n]| = 4 \times |0{,}625| + 2 \times 1{,}25 + 2 \times 5 = 15$

Si può, al peggio, avere una crescita di:

$\log_2 15 = 4$ bit

c) Disegno la diretta singola: $y = ax[n] + bx[n-1]$

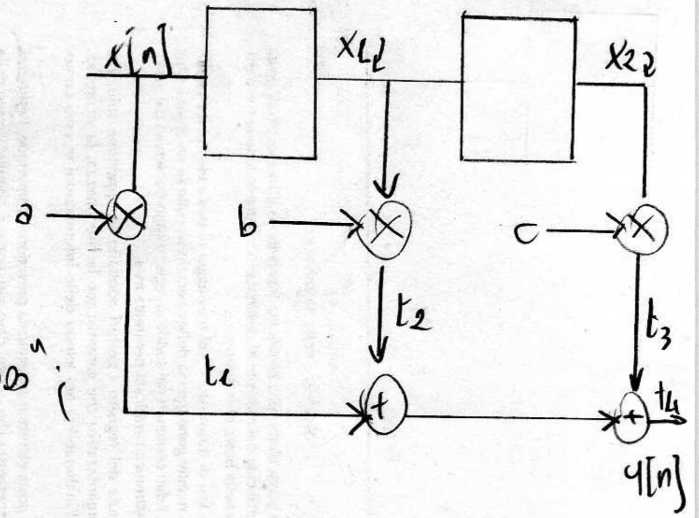


1) Scambio x e y
2) Scambio (+) e ⟂
3) Preservo D e (×)
4) Inverto gli archi

Descrizione Behavioural della architettura

Codice

```
ARCHITECTURE Behavioural of FIR IS
SIGNAL x1, x2, t1, t2, t3 : INTEGER;
SIGNAL ytmp : std_logic_vector (N-1 downto 0);
BEGIN
RegProc : PROCESS(CLK, RSTn)
   BEGIN
   IF (RSTn = '0') then x1 <= "000"; x2 <= "000";
   ELSIF (CLK = '1' AND CLK'EVENT) Then
    x1 <= x;
    x2 <= x1;
   END IF;
  END PROCESS;
CombProc : PROCESS(x, x1, x2, t1, t2, t3, t4)
 BEGIN
  t1 <= a * x;
  t2 <= b * x1;
  t3 <= c * x2;
  t4 <= t1 + t2 + t3;
 END PROCESS;
 ytmp = conv_to_std_logic_vector(t4, N);
 y <= ytmp; -- (o un suo subset, per troncare)
 END ARCHITECTURE
```



Idea: ho un process, per il filtro, che genera tutti e soli i registri, dandone segnali di ingresso e uscita. Nell'esempio, x: x[n], x1: x[n-1], e così via.

In un secondo process si gestirà le parte combinatoria.

Un suggerimento è: far tutto con gli integer, e a posteriori convertire nel formato desiderato.

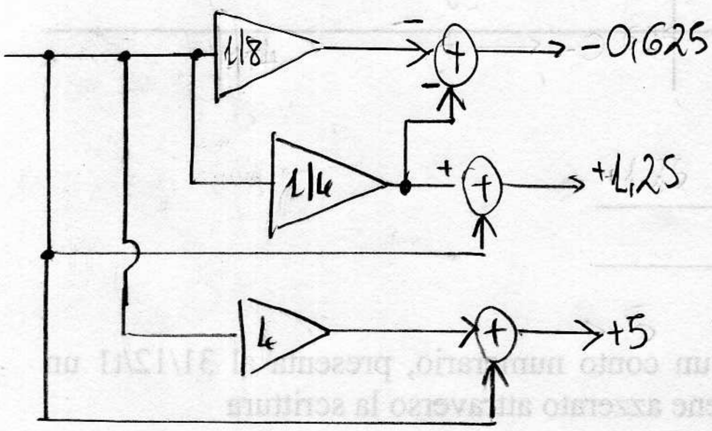
(T4)

e) Si fornisce, a partire dalle 5 regole, ciò che è richiesto.

1) $\{-0{,}625 \; ; \; 1{,}25 \; ; \; 5\}$;

2) $\{0{,}625 \; ; \; 1{,}25 \; ; \; 5\}$

~~3)~~ 4) $\{2^{-2}+2^{-3} \; ; \; 1+2^{-2} \; ; \; 1+2^{2}\}$



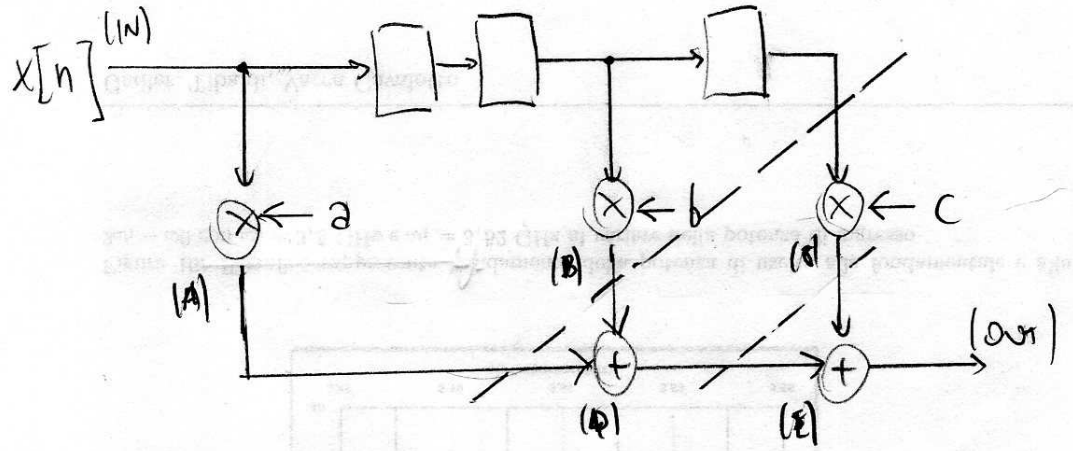
f) Nel caso della forma trasposta, ho $T_{CK} = T_m + T_a$;

$$\boxed{T_h = \frac{1}{T_m + T_a}}$$

Questo perché il filtro produce un campione per volta.

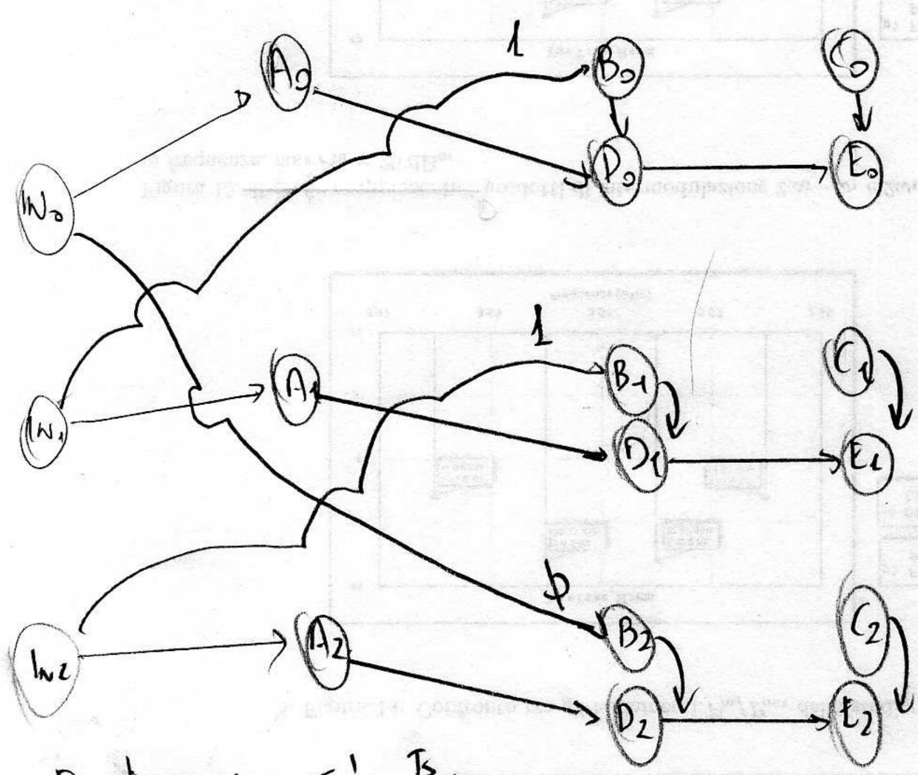
3) $y[n] = a\,x[n] + b\,x[n-2] + c\,x[n-3]$ (T5)

a)



basta aggiungere 1 registro per ogni intersezione.

b)



$j = (w+i)\,\%\,3 = 2$

$IN_0 \to B$: $w_j = 0$

$IN_1 \to B$: $j = 0$, $w_j = 1$

$IN_2 \to B$: $j = 1$, $w_j = 1$

$IN \to C$: 3 registri !

$j = i$

$w_j = \frac{w}{J} = \frac{w}{3} = 1.$

Prestazioni : $T_s' = \frac{T_s}{3}$;

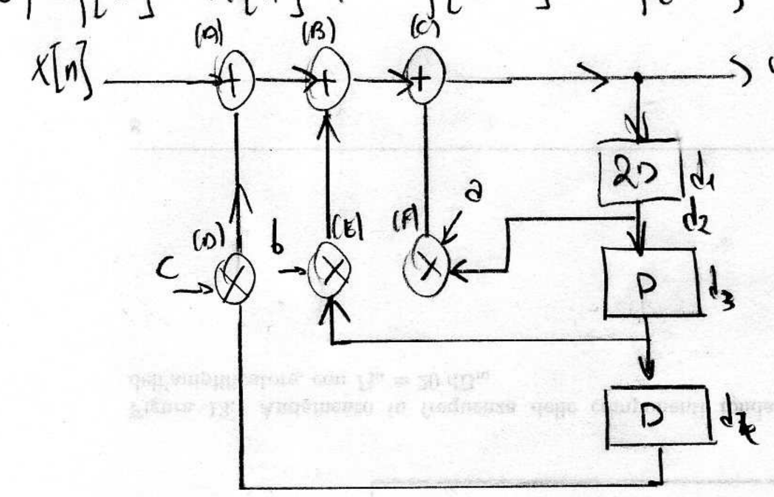
a ogni colpo prendo 3 campioni ; T_{CK} è uguale a prima.

C : ?? Come si fa ad aver $T_{CK} = T/k$? Grano fine ?

Ricordo alcune definizioni:

- T_{CP}: tempo di critical path: ritardo del cammino peggiore;
- T_S: tempo di campionamento;
- throughput: numero di campioni prodotti x unità di tempo;

6) $y[n] = x[n] + a\,y[n-2] + b\,y[n-3] + c\,y[n-4]$ $\quad T_a = 1$, $T_m = 4$

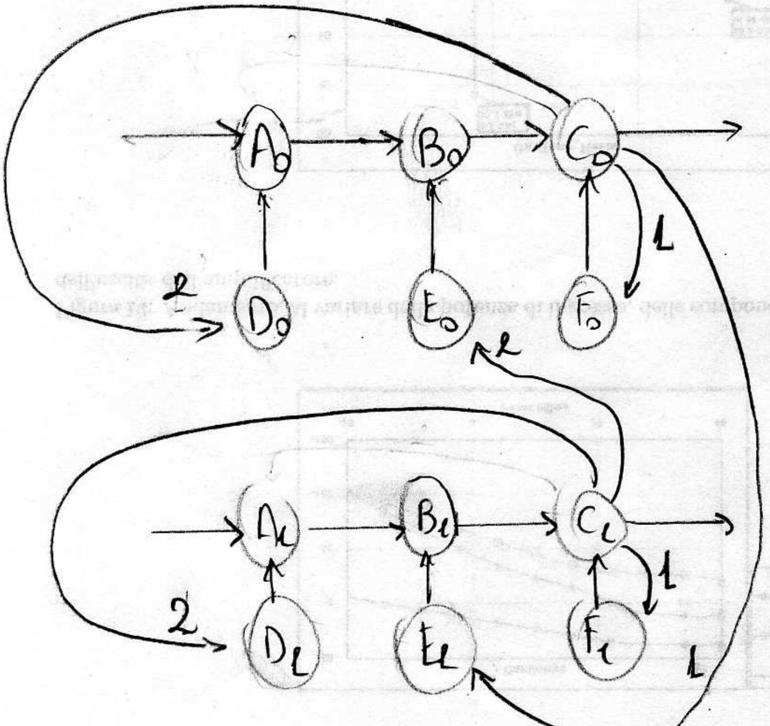


Determinare T_∞ e T_{cp}.

T_{cp}: dal registro d_4 a $y[n]$ (non è indipendente dalla struttura: se lavoro $x[n] + a\,y[n-2]$ invece di $x[n] + c\,y[n-4]$ cambia).

$$T_\infty:\ \max\left\{\frac{T_a+T_a}{2},\ \frac{3T_a+T_m}{4},\ \frac{T_m+2T_a}{3}\right\} = \max\left\{1,\ \frac{7}{4},\ 2\right\} = 2$$ (OK ??

ome



$C_0 \to F$ $\begin{cases} j = (0+2)\%2 = 0 \\ w_j = 1 \text{ (salto)} \end{cases}$

$C_0 \to D$ uguale.

$C_0 \to E$: $\begin{cases} j = (0+3)\%2 = 1 \\ w_j = (0+3)/2 = 1 \end{cases}$

$C_1 \to E$: $\begin{cases} j = (1+3)\%2 = 0; \\ w_j = 2 \end{cases}$

$$\underline{\underline{L}}^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 5 & -1 & 0 & -1 \\ 6 & -1 & -1 & 0 \\ 7 & -1 & -1 & -1 \end{bmatrix}$$

$T_{CK} = T_{CK,old} = \dfrac{1}{\underbrace{4+3}_{(T_m+3T_a)}}$; $\quad T_h = \dfrac{2}{T_m+3T_a}$

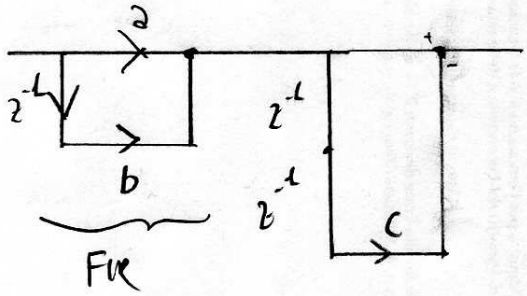
$$\underline{\underline{L}}^{(2)} = \begin{bmatrix} 5 & -1 & 0 & -1 \\ 6 & 5 & -1 & 0 \\ 7 & 6 & -1 & -1 \\ -1 & 7 & -1 & -1 \end{bmatrix} \quad \underline{\underline{L}}^{(3)} = \begin{bmatrix} 6 & 5 & -1 & 0 \\ 10 & 6 & 5 & -1 \\ 11 & 7 & 6 & -1 \\ 12 & -1 & 7 & -1 \end{bmatrix} \quad \underline{\underline{L}}^{(4)} = \begin{bmatrix} 10 & 6 & 5 & -1 \\ 11 & 10 & 6 & 5 \\ 12 & 11 & 6 & 6 \\ 13 & 12 & -1 & 7 \end{bmatrix}$$

(T7)

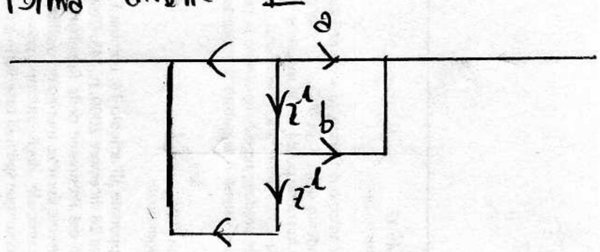
7) $y[n] = \underbrace{a\,x[n] + b\,x[n-1]}_{FIR} + \underbrace{c\,y[n-2]}_{\text{"IIR"}} \xrightarrow{Z} aX + bXz^{-1} + cYz^{-2}$

$\Rightarrow \dfrac{Y}{X} = \dfrac{a + bz^{-1}}{1 - cz^{-2}}$

Forma Diretta I



Forma diretta II

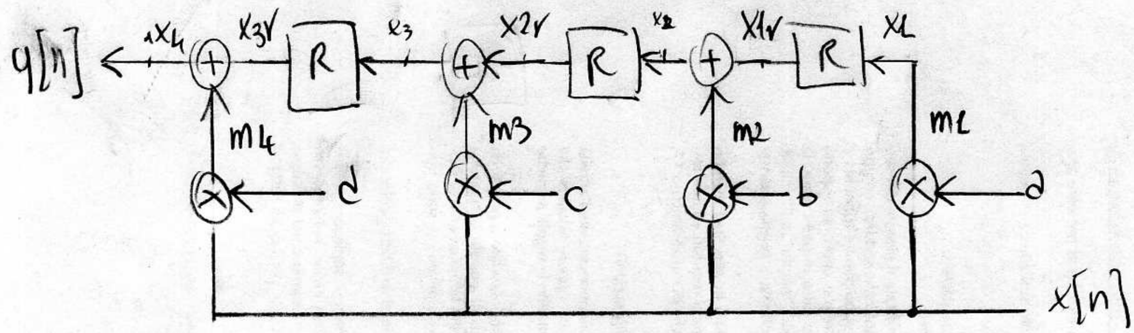


Vantaggi: usa meno registri di prima, senza introdurre particolari complicazioni (scambia: prima parte IIR, poi FIR).

Forma parallela: sarebbe bello con dei numeri.
Vantaggio: fa strutture più semplici. Calcola residui ...

Altra struttura: seriale (metter in catena blocchi): permette anch'essa di usare blocchi più semplici

8)




```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;


ENTITY FIR IS
  PORT ( x :   IN std_logic_vector (3 downto 0);
         CLK, RSTn : IN std_logic;
         y : out std_logic_vector (3 downto 0));
END ENTITY
```

ARCHITECTURE Behavioural OF FIR IS
SIGNAL x1, x1r, x2, x2r, x3, x3r, m1, m2, m3, m4, x4 : INTEGER;

BEGIN
SeqPart: PROCESS(CLK, RSTn)
BEGIN
IF RSTn='0' then x1r='0'; x2r='0'; x3r='0';
ELSIF (CLK'EVENT AND CLK='1') THEN
x1r <= x1; x2r <= x2; x3r <= x3;
END IF;
END PROCESS;
CombPart: PROCESS(x1, x2, x3, x1r, x2r, x3r, m1, m2, m3, m4, x)
BEGIN
m1 <= a * x; x2 <= m2 + x1r;
m2 <= b * x; x3 <= m3 + x2r;
m3 <= c * x; x4 <= m4 + x3r;
m4 <= d * x;
-- Spero funzioni;
y <= conv_to_std_logic_vector(x4, L);
END PROCESS;
END ARCHITECTURE;

L'idea dell'arch. distribuita è: dato $y = \sum_{n=0}^{N-1} c[n]\,x[n]$, posso trattare $x[n]$ come:

$$x[n] = \sum_{b=0}^{B-1} x_b[n]\,2^b$$

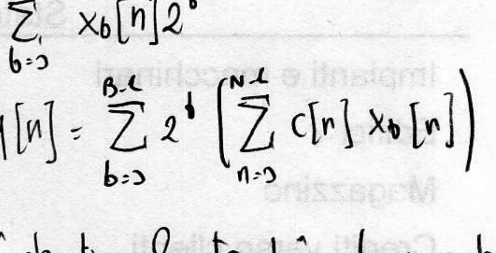
$$\underset{\text{distributivita}}{\rightarrow}\; y[n] = \sum_{b=0}^{B-1} 2^b \left[\sum_{n=0}^{N-1} c[n]\,x_b[n]\right]$$

Ma, "dentro la tonda", b è costante; varia nella sommatoria "esterna".

Realizzazione in aritmetica distribuita: Questo fatto permette di trattare y come un insieme di valori di una funzione f, i cui valori son immagazzinati in una MEMORIA, i cui indirizzi sono gli $x_b[n]$: a seconda dei valori di x_b che si hanno, si avran in uscita diversi f.

Vantaggio: i campioni son precalcolati, dunque la complessità sarà minore. Ci vorran B cicli, con B numero di bit, per aver y in uscita.

Questa particolare implementazione, poi, non richiede barrel shifters, quindi è particolarmente semplice.



9) $T_{cp} = 3 \begin{pmatrix} A \to B \to C \\ (e\ A \to F) \end{pmatrix}$; Io metterei un cut su $\begin{pmatrix} A \to B \\ e \\ A \to F \end{pmatrix}$, e uno su $\begin{pmatrix} G \to E \\ e \\ D \to E \end{pmatrix}$; (T9)

NON mettendolo, posso spostare C→D a B→C;

Provo un metodo più formale:

arco	w(e)	$w_r(e)$	t(u)+t(v)
A→B	0	$1 + r(A) - r(B)$	2
A→F	0	$0 + r(A) - r(F)$	3 → crit.
B→G	1	$1 + r(B) - r(G)$	2
B→C	0	$0 + r(B) - r(C)$	2
C→D	2	$2 + r(C) - r(D)$	2
F→D	1	$1 + r(F) - r(D)$	3 → crit.
G→E	0	$0 + r(G) - r(E)$	2
D→E	0	$0 + r(D) - r(E)$	2

$$\begin{cases} 1 + r(A) - r(B) \geq 0 \\ r(A) - r(F) \geq 1 \\ r(B) - r(G) \geq -1 \\ r(B) - r(C) \geq 0 \\ r(C) - r(D) + 2 \geq 0 \\ \cancel{1} + r(F) - r(D) \geq \cancel{1} \\ 0 + r(G) - r(E) \geq 0 \\ 0 + r(D) - r(E) \geq 0 \end{cases}$$

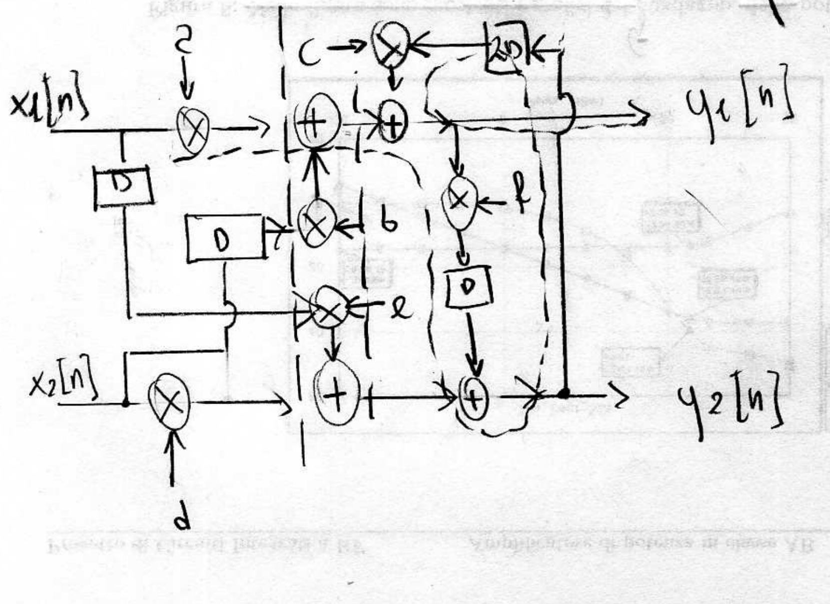
$\boxed{\text{Fisso}}$ $r(E) = 0$; $r(G) = 0$; $r(B) = -1$; $r(D) = 0$; $r(A) \geq 0$; → $r(A) = 1$; $r(F) = 0$;

$r(G) \geq 0$
$r(D) \geq 0$
$r(F) \geq r(D)$;
$r(C) \geq r(D) - 2$;
$r(B) \geq r(C)$;
$r(B) \geq r(G) - 1$;
$r(A) \geq r(F) + 1$;
$r(A) \geq r(B) - 1$

$$\begin{cases} r(A) = 1; \\ r(B) = -1; \\ r(C) = -1; \\ r(D) = 0; \\ r(E) = 0; \\ r(F) = 0; \\ r(G) = 0; \end{cases}$$

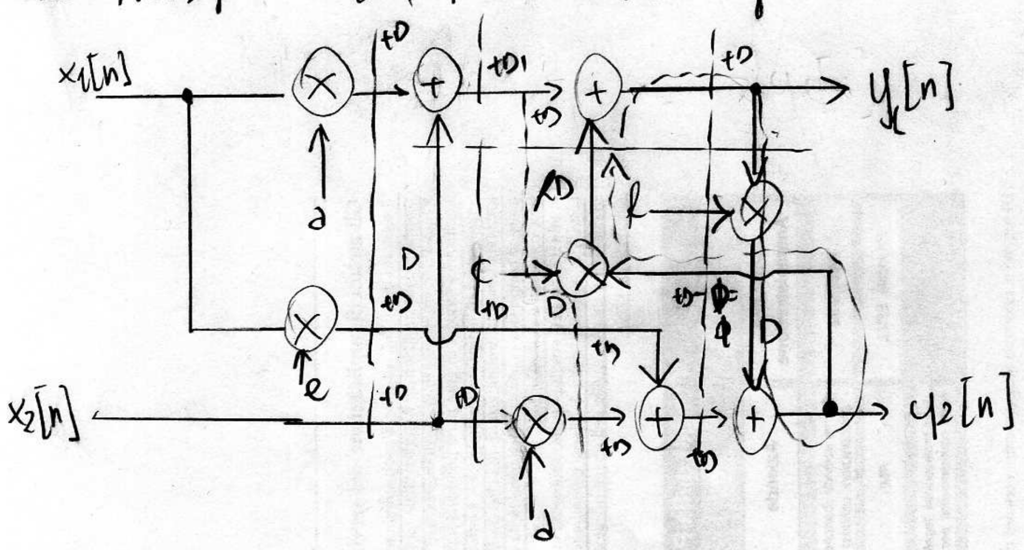
?

10)



Applico i 2 tagli e poi dovrei fare retiming

Lo ridisegno meglio, per fare pipelining: T10



$T_{cp} = 2T_m + 2T_a = 10$

$T_\infty = \frac{1}{3}\left(T_a + T_m + T_a + T_m\right)$

$= \frac{1}{3}(10) = 3{,}3$

A 3,3 non posso arrivare: al min. 4 (T_m).

Look ahead:

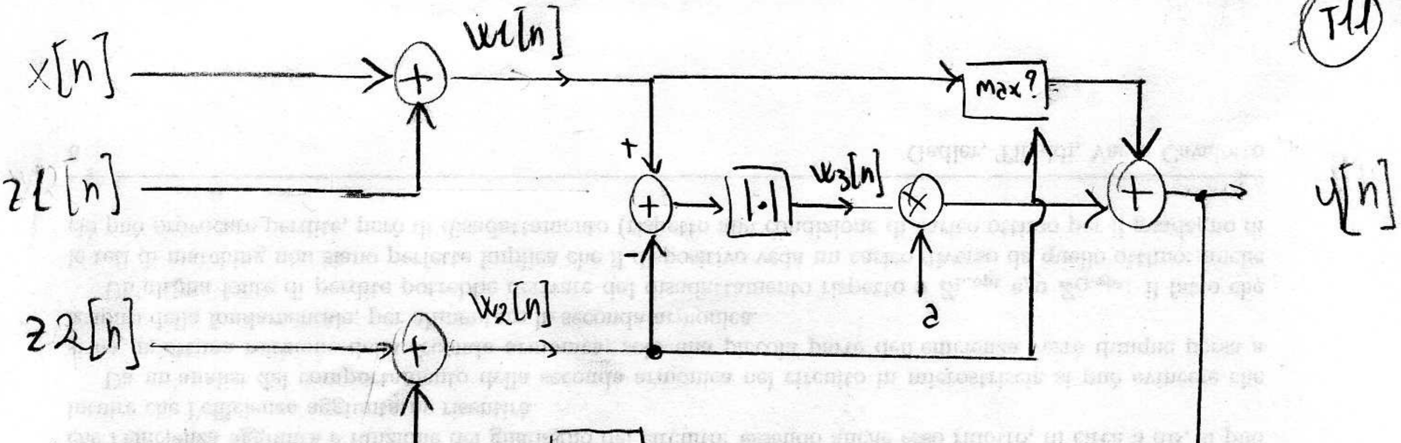
$y_1[n+1] = a\,x_1[n+1] + b\,x_2[n] + c\,y_2[n-1]$

$y_2[n+1] = d\,x_2[n+1] + e\,x_1[n] + f\,y_1[n] = d\,x_2[n+1] + e\,x_1[n] + f\big(a\,x_1[n] + b\,x_2[n-1] + c\,y_2[n-2]\big).$

?

Enorme! Bar

11) T11

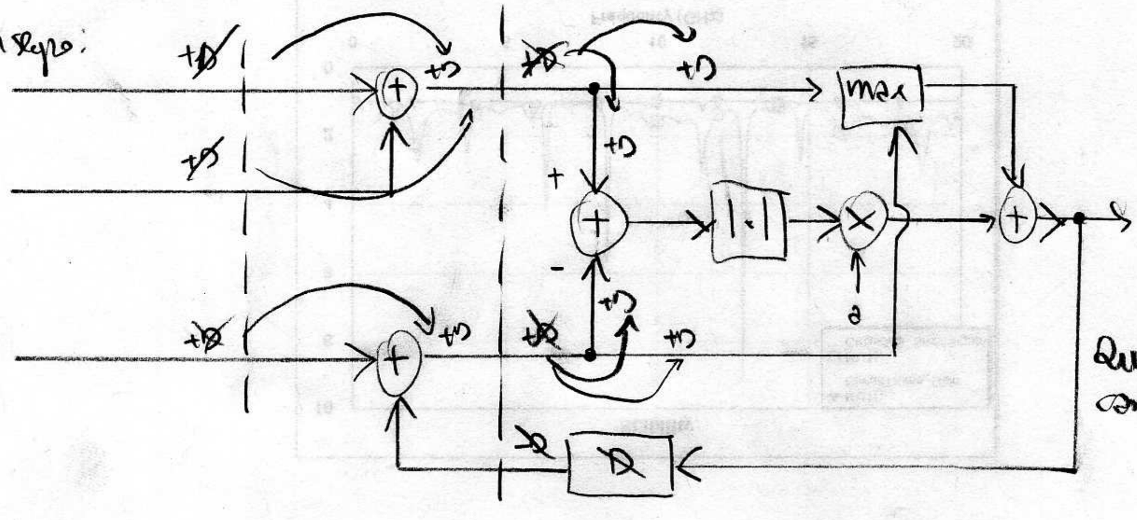


$T_a = 2$; $T_{max} = 3$; $T_{abs} = 4$; $T_{mul} = 5$

$T_{cp} = 2T_a + T_{abs} + T_{mul} + T_a = 15$

$T_\infty = 3T_a + T_{abs} + T_{mul} = 15$

Ridisegno:



Qua T_∞ è cambiato...

Risultato "finale":