UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Arquitecturas Paralelas

Procesado Digital de la Señal en FPGA

Máster Universitario en Ingeniería de Sistemas Electrónicos

Objetivos

- Diseñar e implementar filtros FIR sobre FPGAs de Xilinx y Altera, usando **arquitecturas paralelas** aprovechando los recursos hardware que ofrecen estos dispositivos

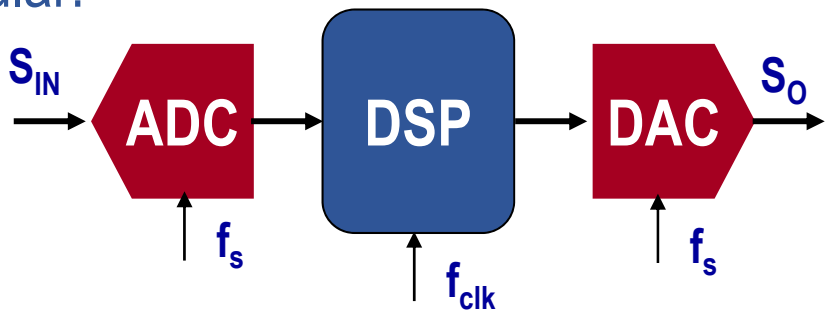
Contenido

- Introducción a las arquitecturas HW
- Arquitecturas paralelas
- Cuantificación de los filtros FIR
- Segmentación y Retiming
- Procesado entrelazado
- Paralelización de algoritmos

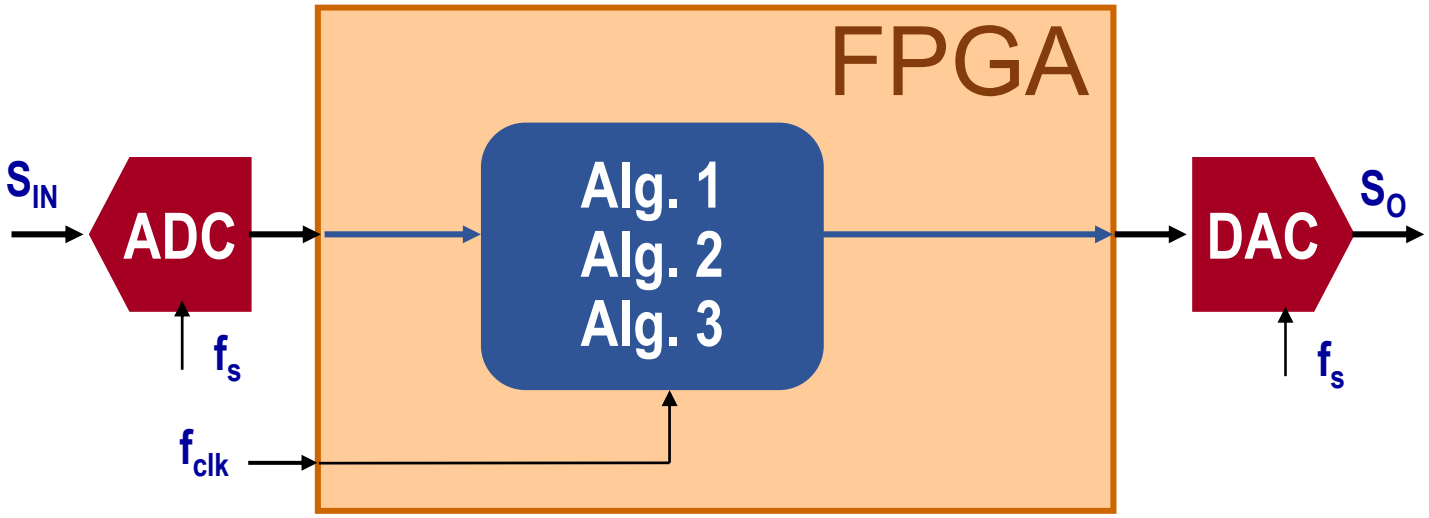
Arquitecturas HW

Ej: Un sistema DSP tiene que calcular:

- Alg.1 (N1 mults)
- Alg.2 (N2 mults)
- Alg.3 (N3 mults)



Caso 1: fclk/fs > (N1+N2+N3)



Procesado secuencial para algoritmos y operaciones incluidas

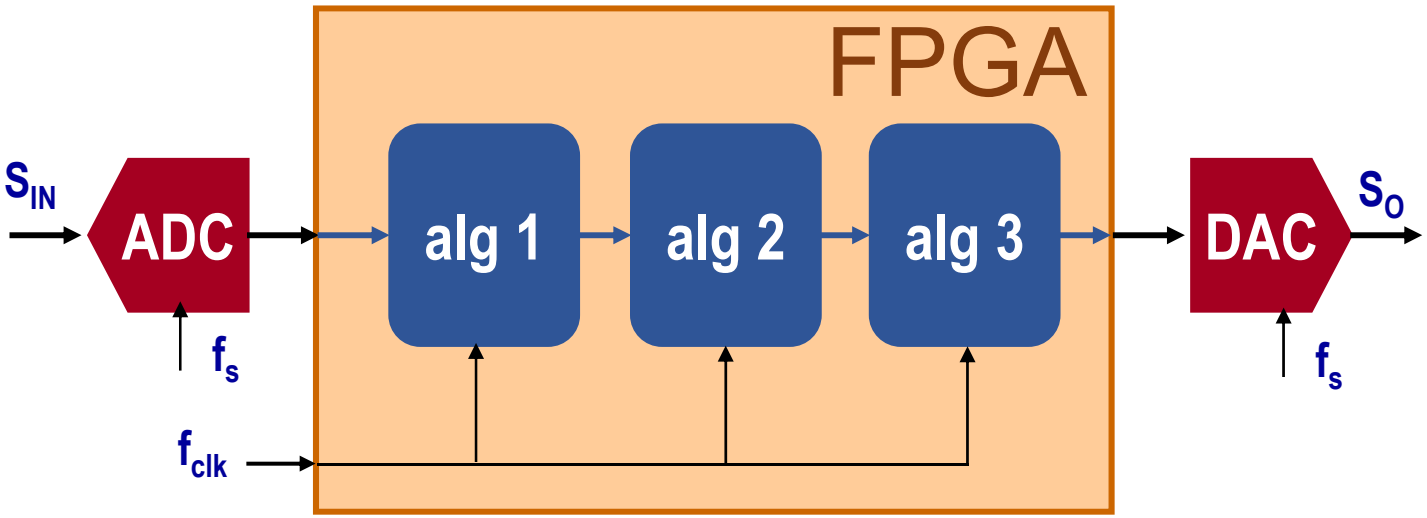
Arquitecturas HW

Caso 2: (N1,N2,N3)< f_{clk}/f_s < (N1+N2+N3)

- Se implementa por separado cada algoritmo
- Segmentación a nivel de bloque
- Arquitectura secuencial para cada algoritmo

Caso 3: $f_{clk}/f_s = 1$

- Se implementa por separado cada algoritmo
- Segmentación a nivel de bloque
- Segmentación interna de los algoritmos para conseguir la f_{clk}

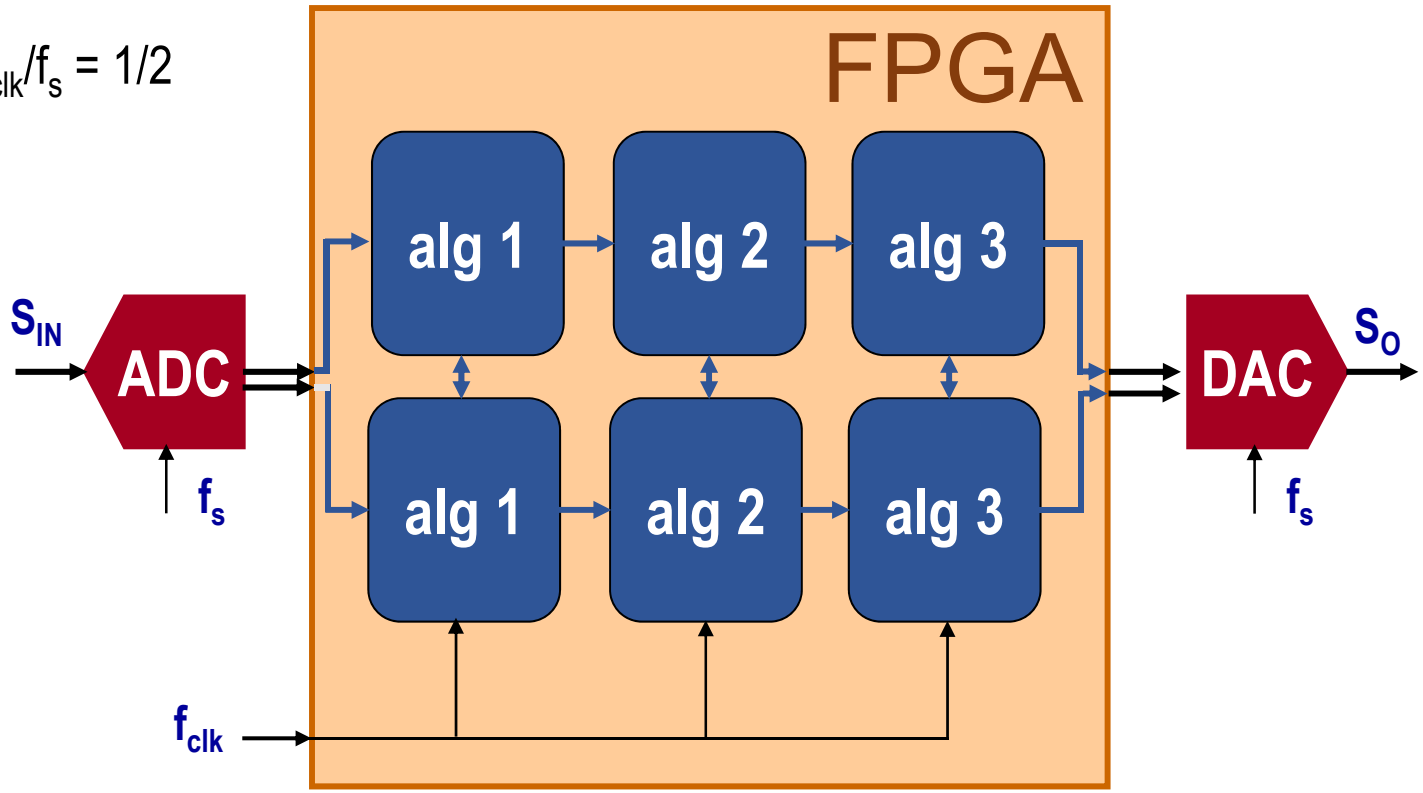


Arquitecturas HW

Caso 4: $f_{clk}/f_s < 1$

- Se implementa por separado cada algoritmo
- Segmentación a nivel de bloque
- Paralelizamos cada algoritmo y segmentamos para alcanzar f_{clk}

Ej: $f_{clk}/f_s = 1/2$



Del algoritmo a la arquitectura

$$y(n) = \sum_{k=0}^{M-1} h_k x(n-k)$$

ALGORITMO

```
y(0) = 0
for k = 1 to M do
  y(k) = y(k-1)+ h(k)*x(k)
end
y = y(n)
```

DSP

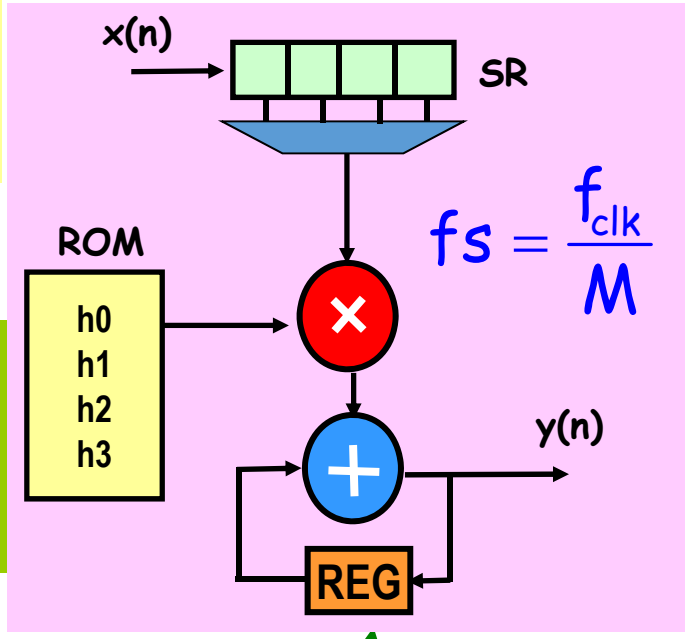
Ex: M=100 $\frac{1\ GHz}{100\ ciclos} = 10\ MSPS$

= FPGA

FPGA

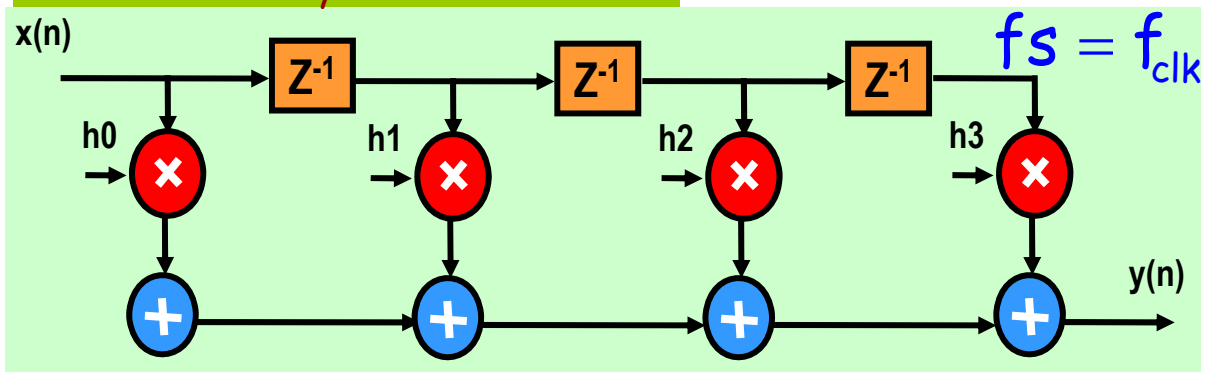
$\frac{400\ MHz}{100\ cycles} = 4\ MSPS$

Folded FIR ≡ Secuencial



Unfolded FIR ≡ Paralelo

FPGA $\frac{100\ MHz}{1\ cycle} = 100\ MSPS$



Area-Throughput

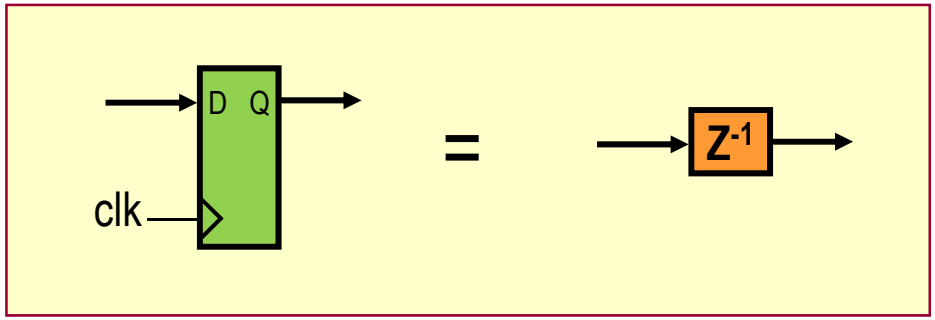
Arquitecturas Paralelas

⇒Frecuencia de muestreo (fs) = Frequencia de reloj (fclk)

⇒La arquitectura necesita tantos recursos como operaciones tiene el algoritmo

⇒Se obtiene un resultado cada ciclo de reloj

⇒Un registro equivale a un retardo de una muestra temporal



Arquitecturas paralelas

Modelo Matlab de un Filtro FIR

Filtro FIR de M-etapas

$$y(n) = \sum_{k=0}^{M-1} h_k x(n-k)$$ ⟵ 1 clock cycle

Modelo Matlab :

```
y(i)=x(i)*h(j)+x(i-1)*h(2) + ··· + x(i-(M-1)*h(M-1);
```

En cada ciclo de reloj:

- Se multiplican M datos por M coeficientes
- Los resultados de las multiplicaciones se suman

Requisitos Hardware :

- M multiplicadores
- M-1 sumadores
- M-1 registros para los datos de entrada
- M registros para los coeficientes

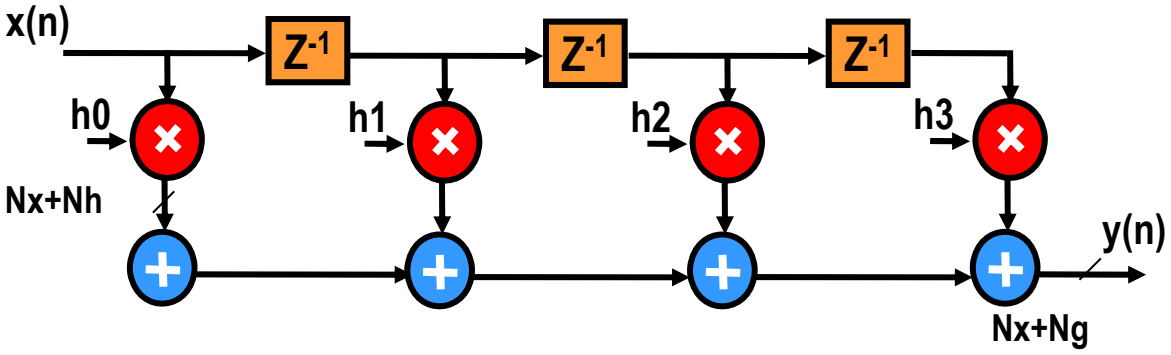
Estructuras para implementar filtros FIR

$$y(n) = \sum_{k=0}^{3} h_k x(n-k)$$

¿Alguna diferencia?

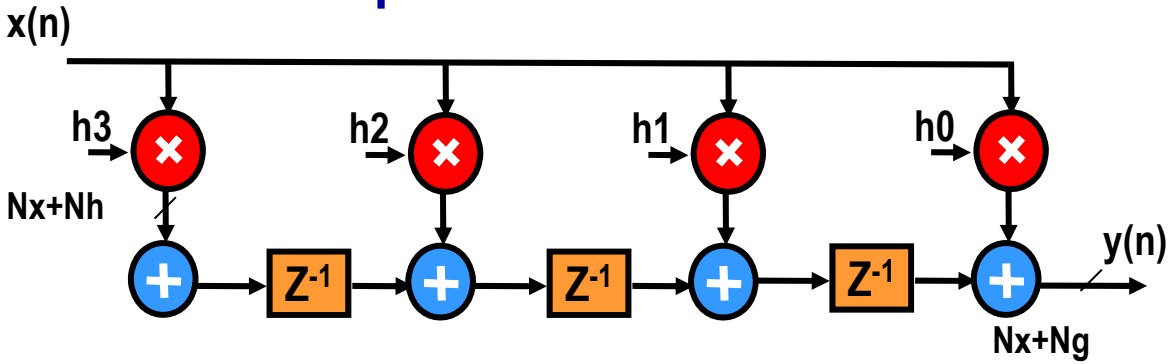
X=Xe2^{-Nx} (Nx bits)
H=He2^{-Nh} (Nh bits)
$P_j = X_i \cdot H_j$ (Nx+Nh bits)
Ng=log$_2$(Ge), Ge=G*2^{Nh}

Forma directa



$t_c = t_{mult} + (M-1)t_{add}$
Nx-bit registers

Forma traspuesta



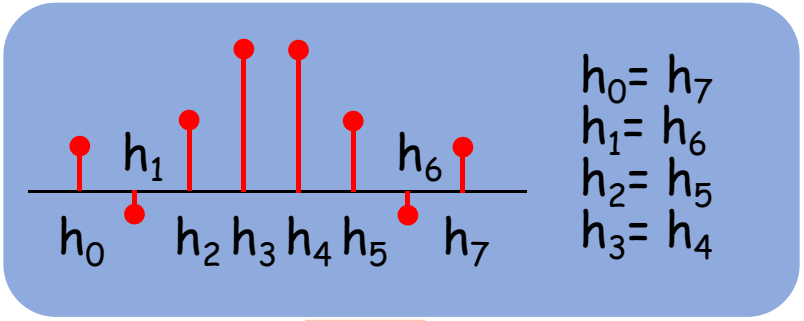
$t_c = t_{mult} + t_{add}$
(Nx+Ng)-bit regist.
Broadcast (↑ fan-out)

Aprovechando las propiedades de los algoritmos

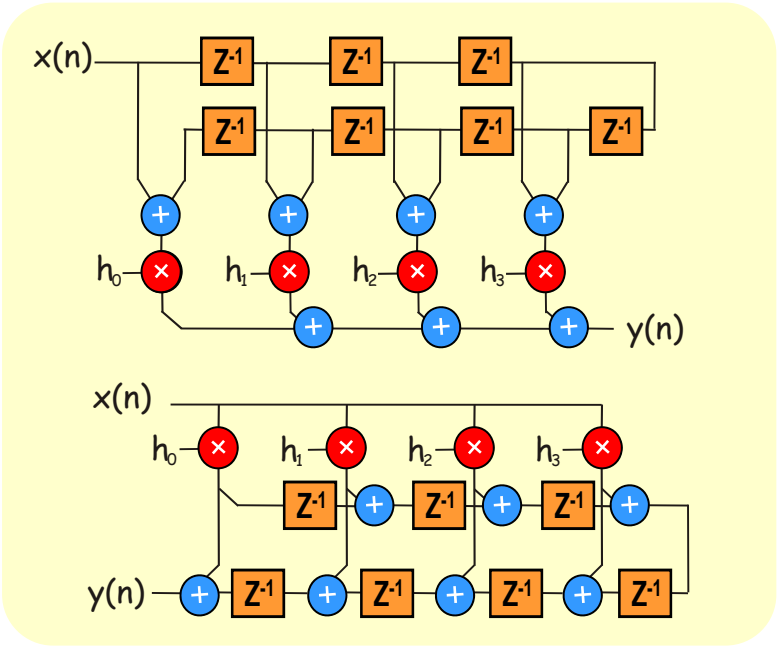
Los filtros FIR de fase lineal poseen **coeficientes simétricos**
Se puede aprovechar para reducir el número de multiplicadores

FIR simétrico de N-etapas (Ej. N=8)

$$y(n)=h_0x(n)+h_1x(n-1)+ h_2x(n-2)+ h_3x(n-3)+ h_4x(n-4)+ h_5x(n-5)+ h_6x(n-6)+ h_7x(n-7)$$



$$y(n)=h_0[x(n)+x(n-7)]+h_1[x(n-1)+x(n-6)]$$
$$+h_2[x(n-2)+x(n-5)]+ h_3[x(n-3)+x(n-4)]$$



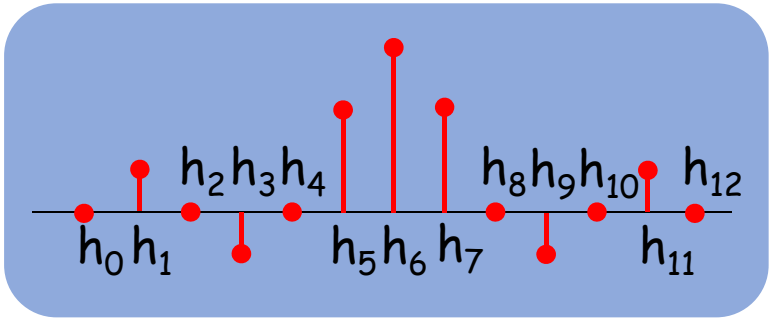
Si el filtro es antisimétrico se cambiarán algunos sumadores por restadores

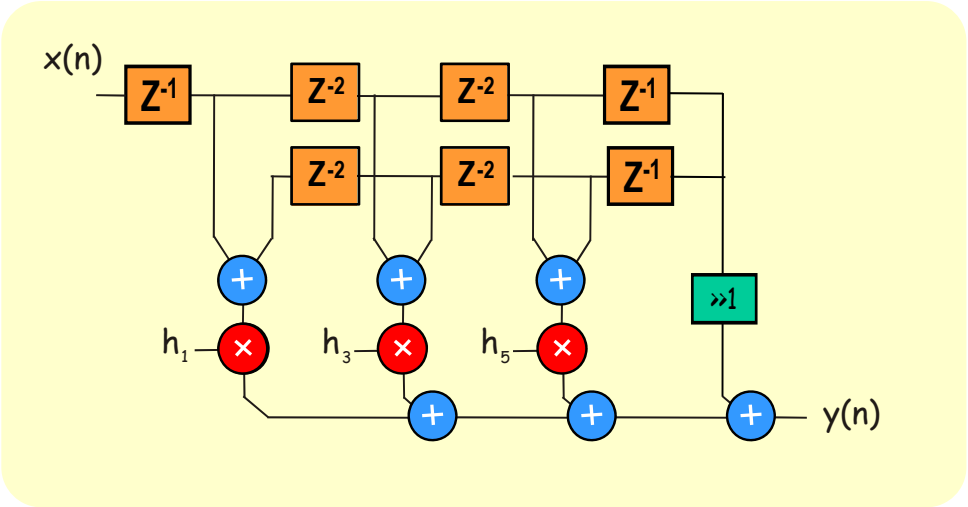
Aprovechando las propiedades de los algoritmos

Algunos filtros poseen coeficientes nulos
→ no es necesario implementar esos mult.

$h_1 = h_{11}$ $h_3 = h_9$ $h_5 = h_7$
$h_6 = 0.5$
$h_0 = h_{12} = 0$ $h_2 = h_{10} = 0$ $h_4 = h_8 = 0$

Filtro MediaBanda

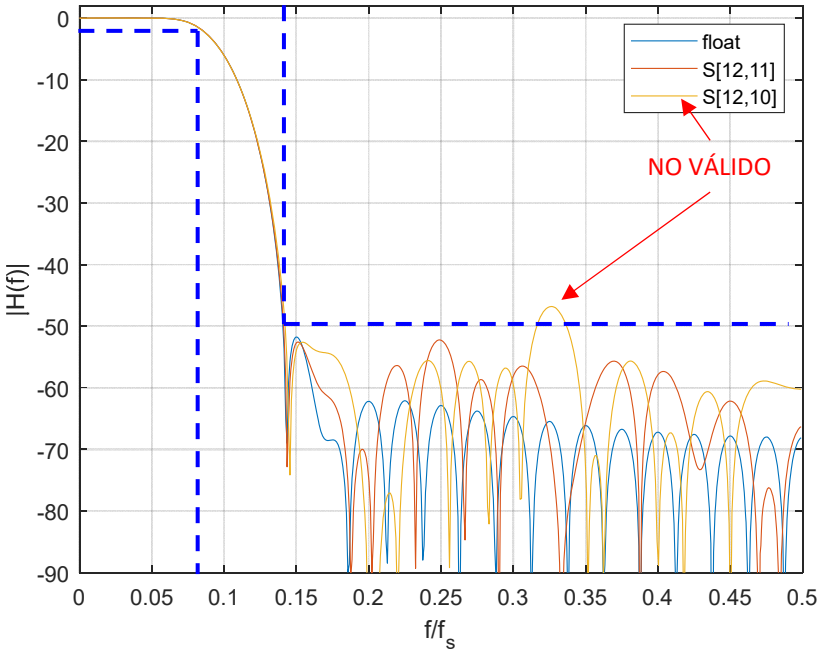




Precisión finita en filtros FIR

Coeficientes:

- Afecta al valor de los ceros
  - ✓ Modifica la respuesta en frecuencia
- Reducción de precision válida mientras se cumpla la mascara de filtrado
- Utilizar “round” → mayor precisión
- Verificación HDL: Golden Model con coeficientes con precisión finta

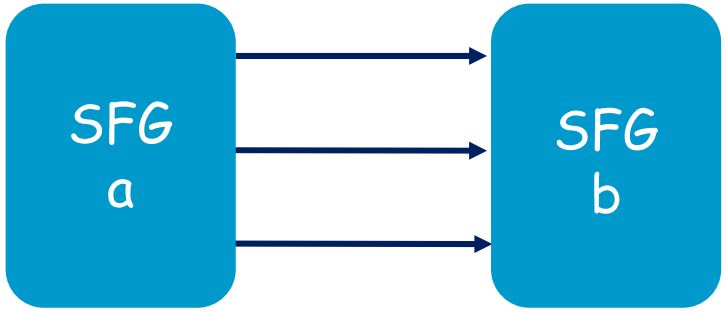


Ruta de datos:

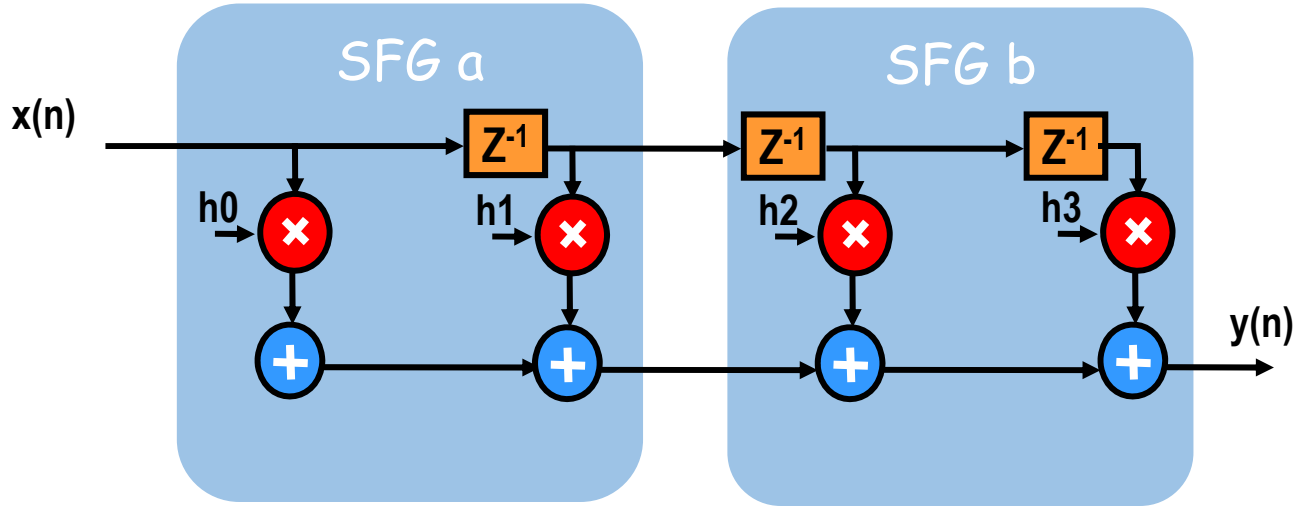
- Introduce ruido de cuantificación
- Usualmente, recorte de precision solo a la salida del filtro
  - ✓ Los bloques DSP de los FPGA: mult + sumador de ancho grande
    - Xilinx Virtex DSP48: 25x18bits + 48 acc
    - Altera Cyclone V Variable DSP: 18x18bits + 44 acc
- Se suele usar “floor” para no incrementar recursos
  - ✓ Algunos bloques DSP posibilitan la implementación de redondeo final

Segmentando los flujos de datos

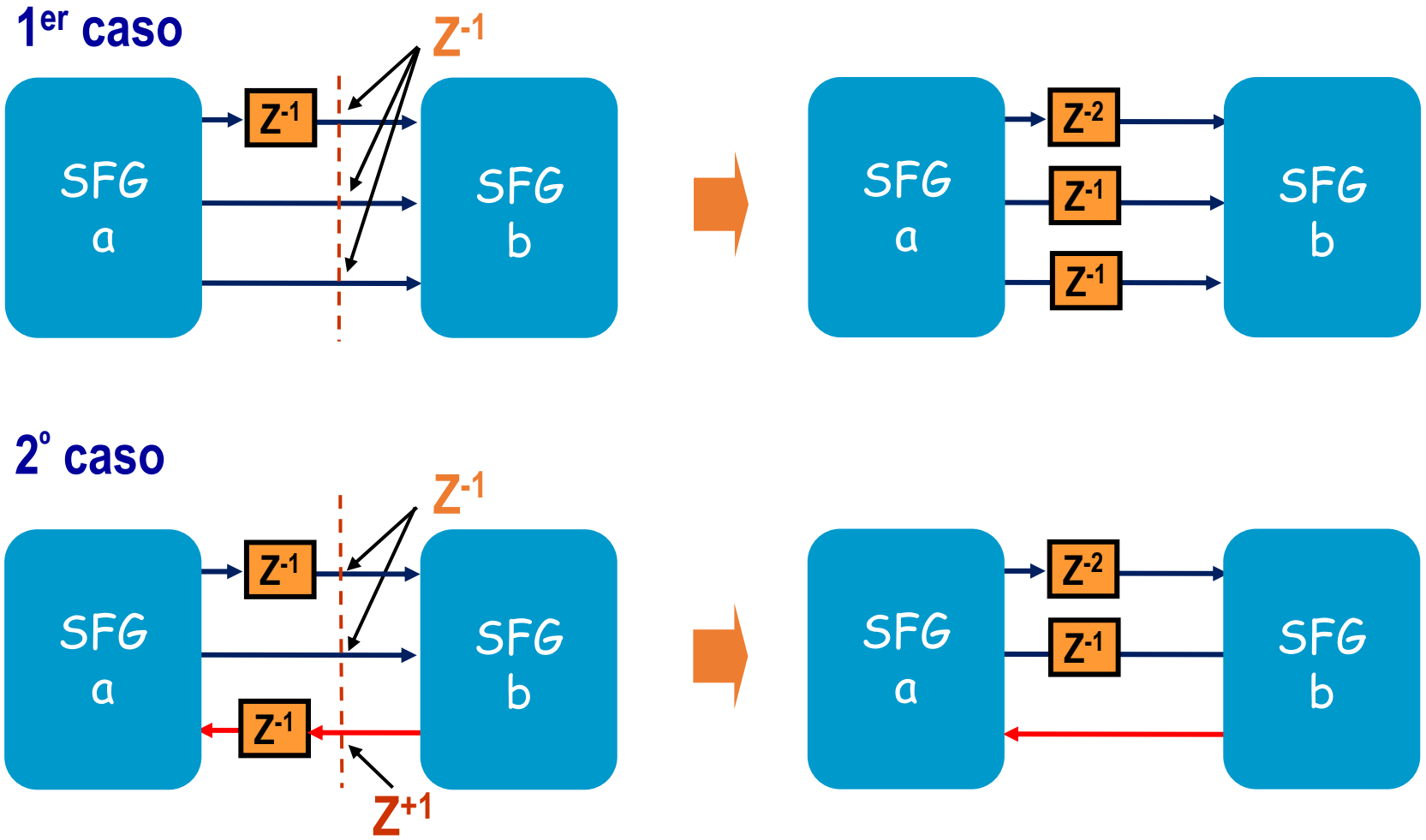
Grafo de flujo de datos



Ejemplo:



Segmentando los flujos de datos



Retiming

Cambiar la ubicación de registros sin modificar el algoritmo

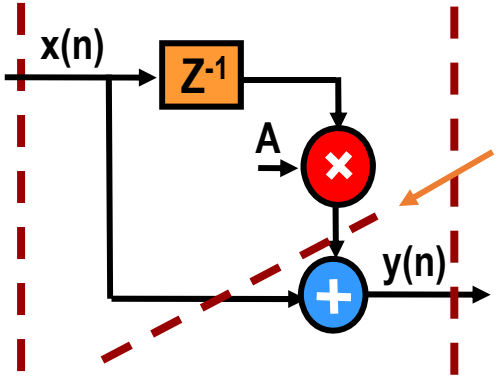
$\Rightarrow$ Transformar la arquitectura $\rightarrow$ adaptarla a bloques DSP

$\Rightarrow$ Reducir el camino crítico

Ej. Reducción del camino crítico:

$y(n)=x(n)+A\cdot x(n-1)$

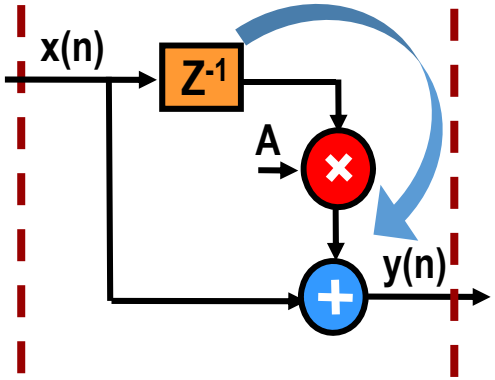
$T_p=t_{mult}+t_{sum}$

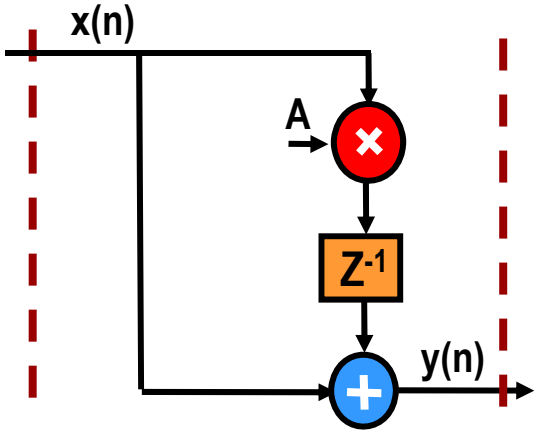


Pipelining

- 2 registros
- añade 1 ciclo de latencia

Retiming





$T_{p1}=t_{mult}$

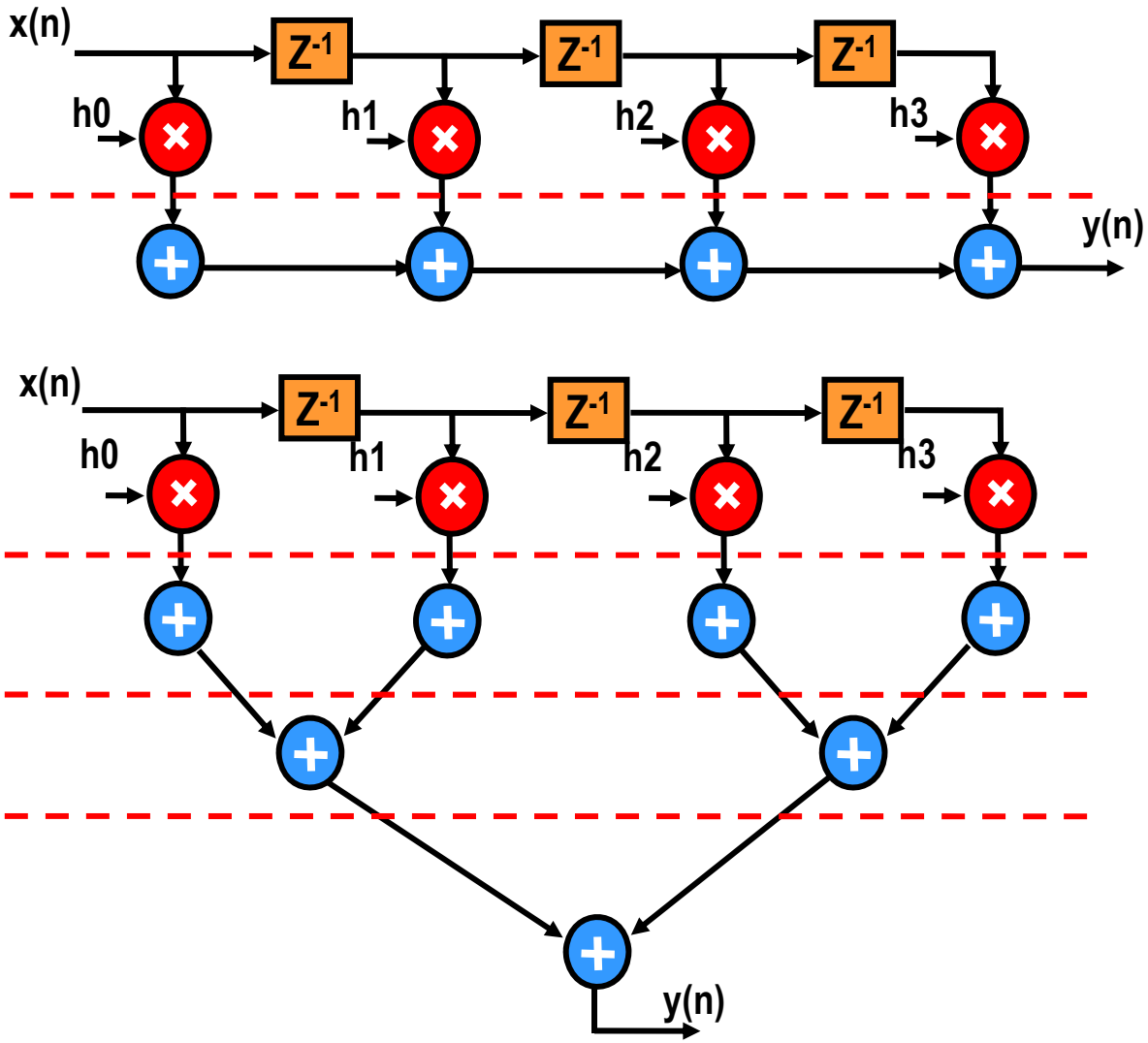
$T_{p2}=t_{sum}$

$T=\max\{T_{p1}, T_{p2}\}=T_{p1}$

¡No aumenta la latencia!

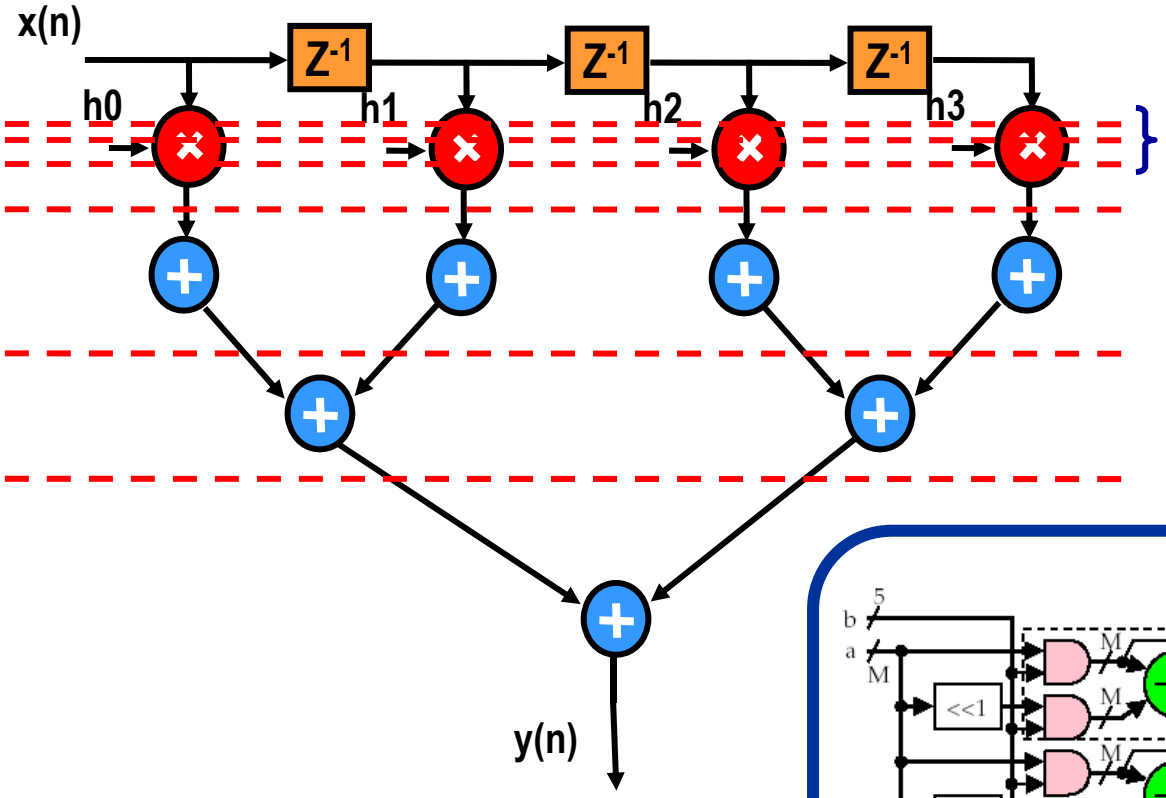
Segmentación en filtros FIR paralelos

Forma Directa

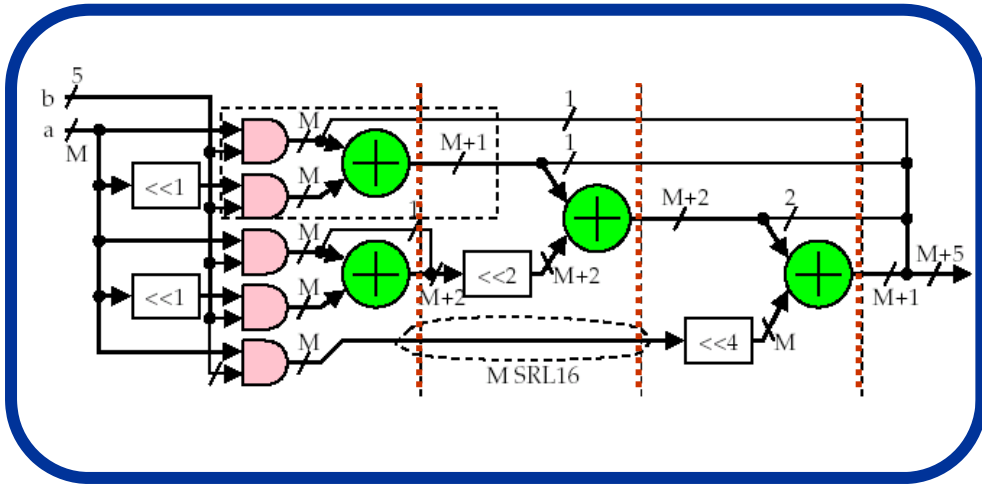


Segmentación en filtros FIR paralelos

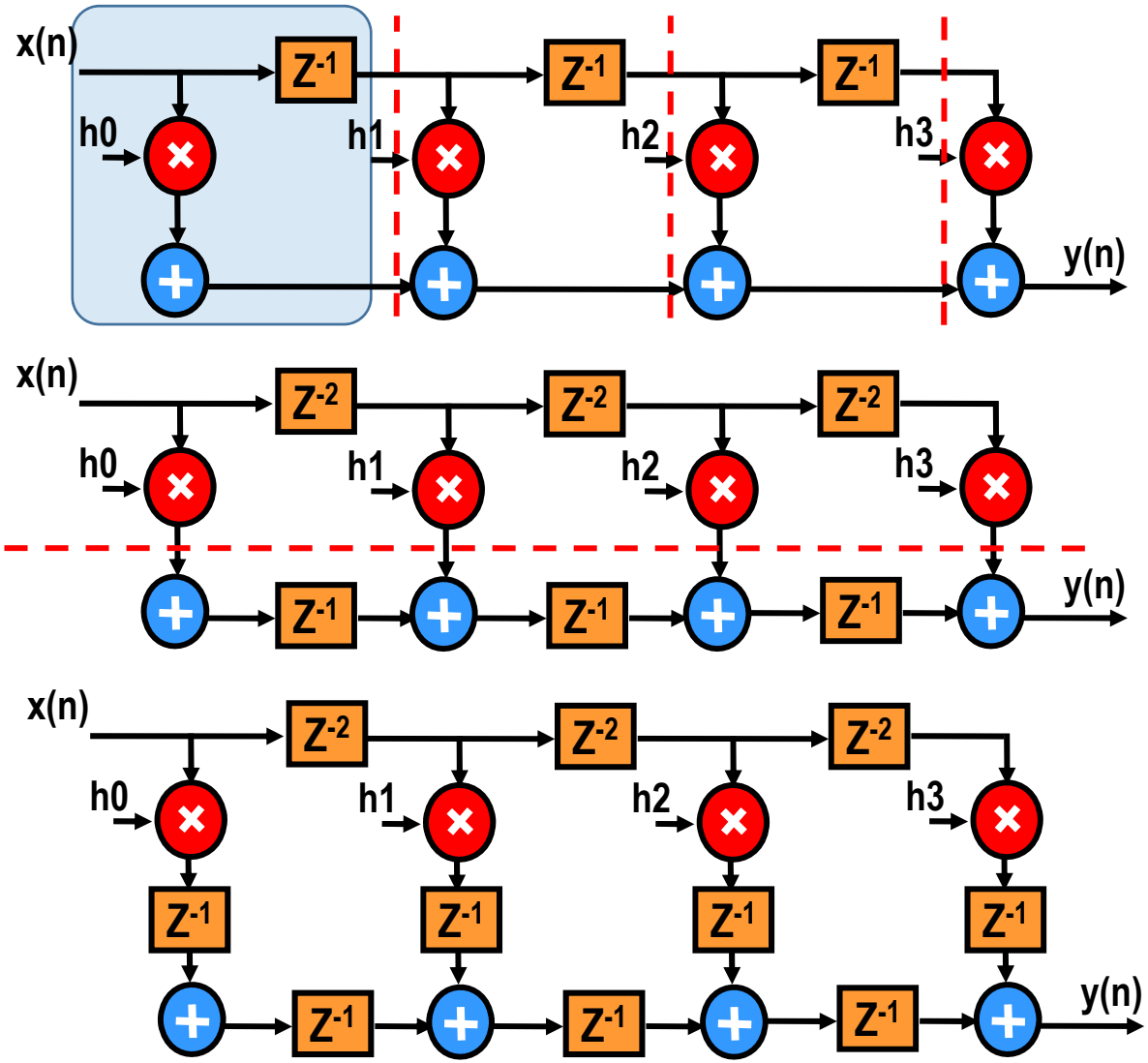
Forma Directa

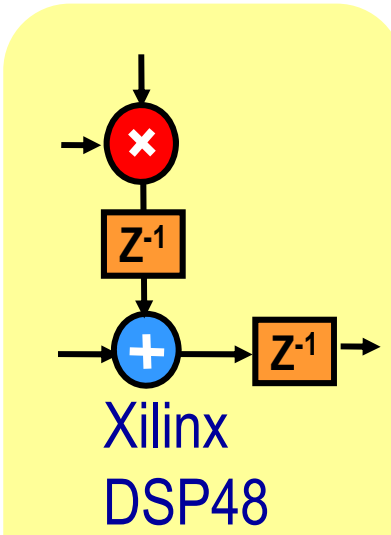


Multiplicadores implementados en LUTs



Segmentación en filtros FIR paralelos

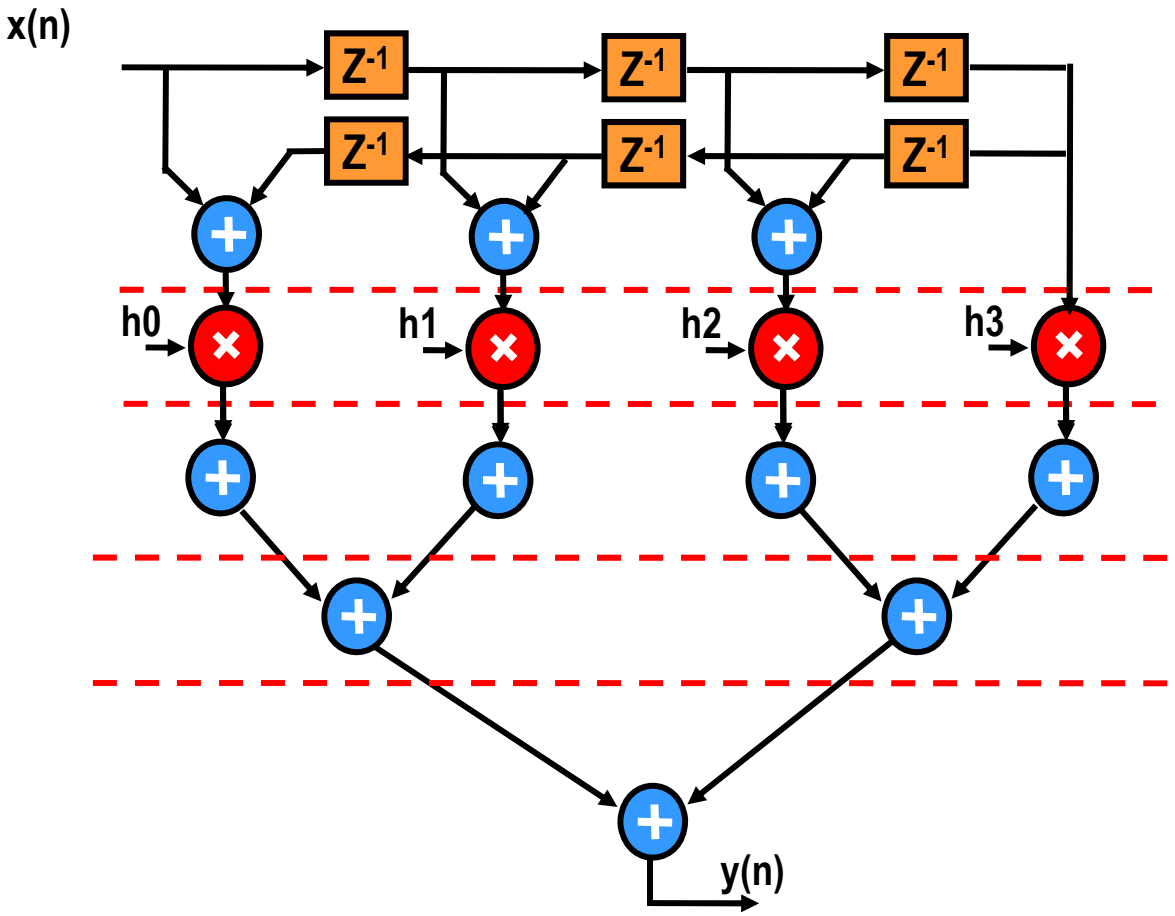




Array sistólico
- Regular
- Modular
- Comunicación local

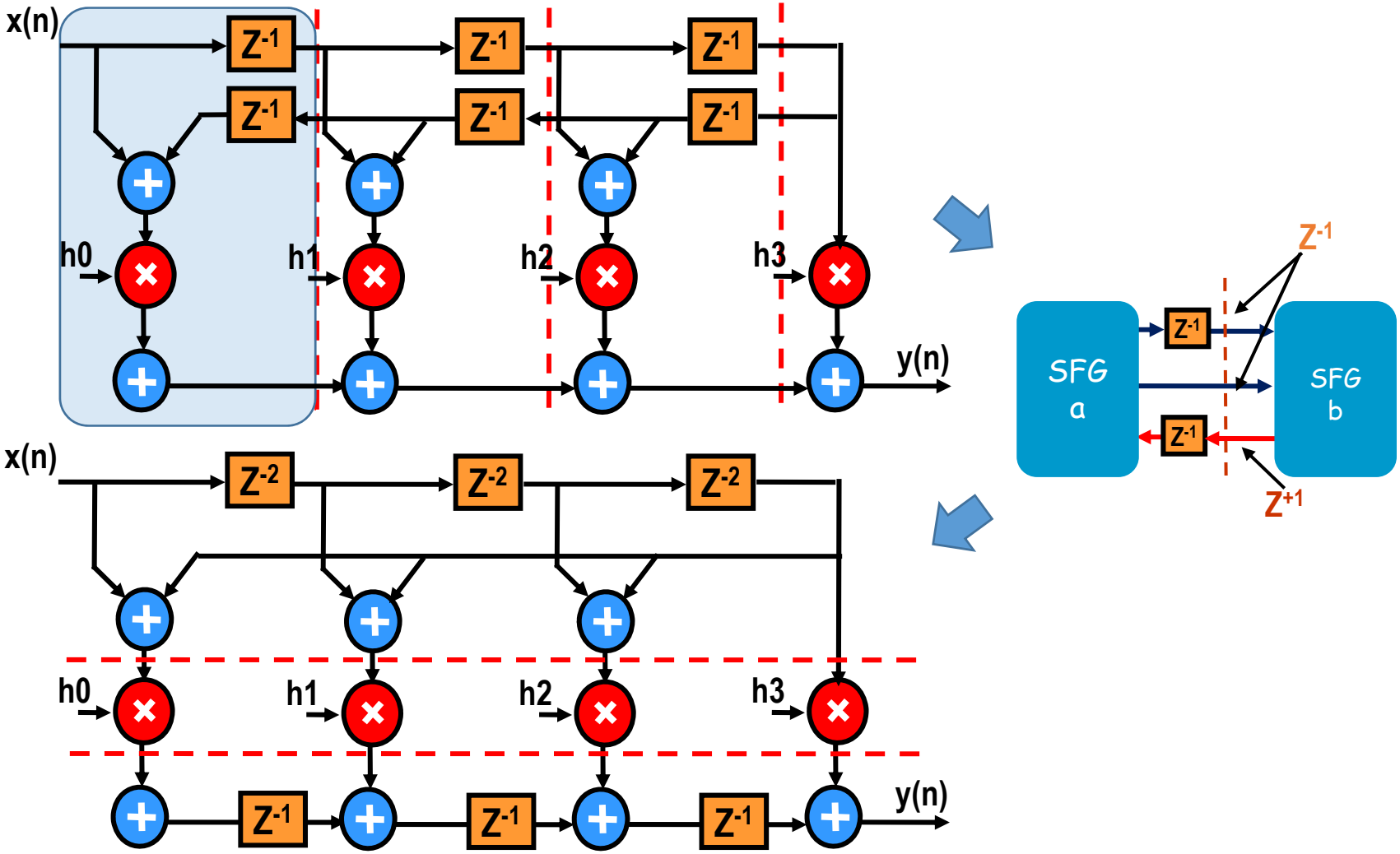
Segmentación en filtros FIR paralelos

FIR Simétrico



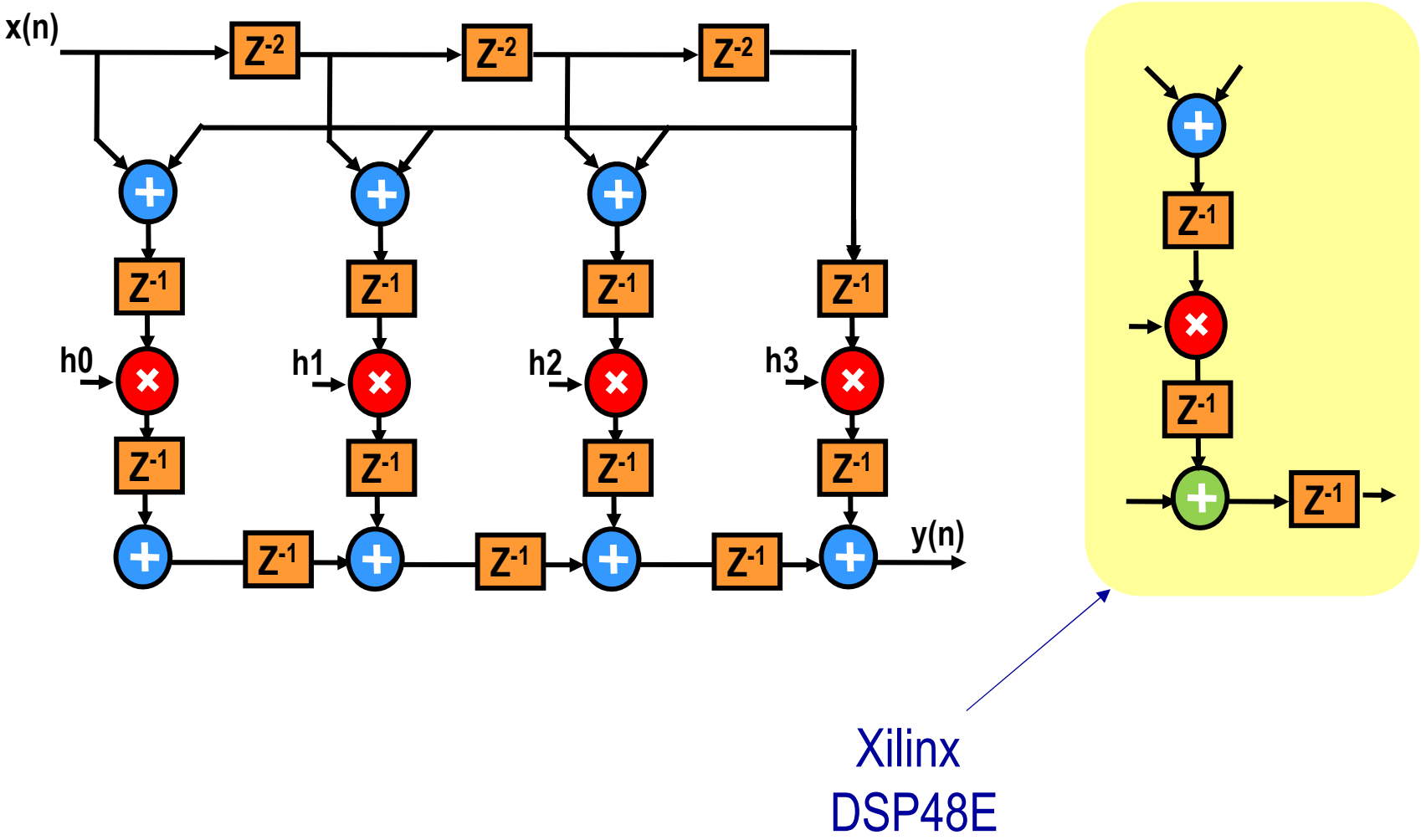
Segmentación en filtros FIR paralelos

FIR Simétrico



Segmentación en filtros FIR paralelos

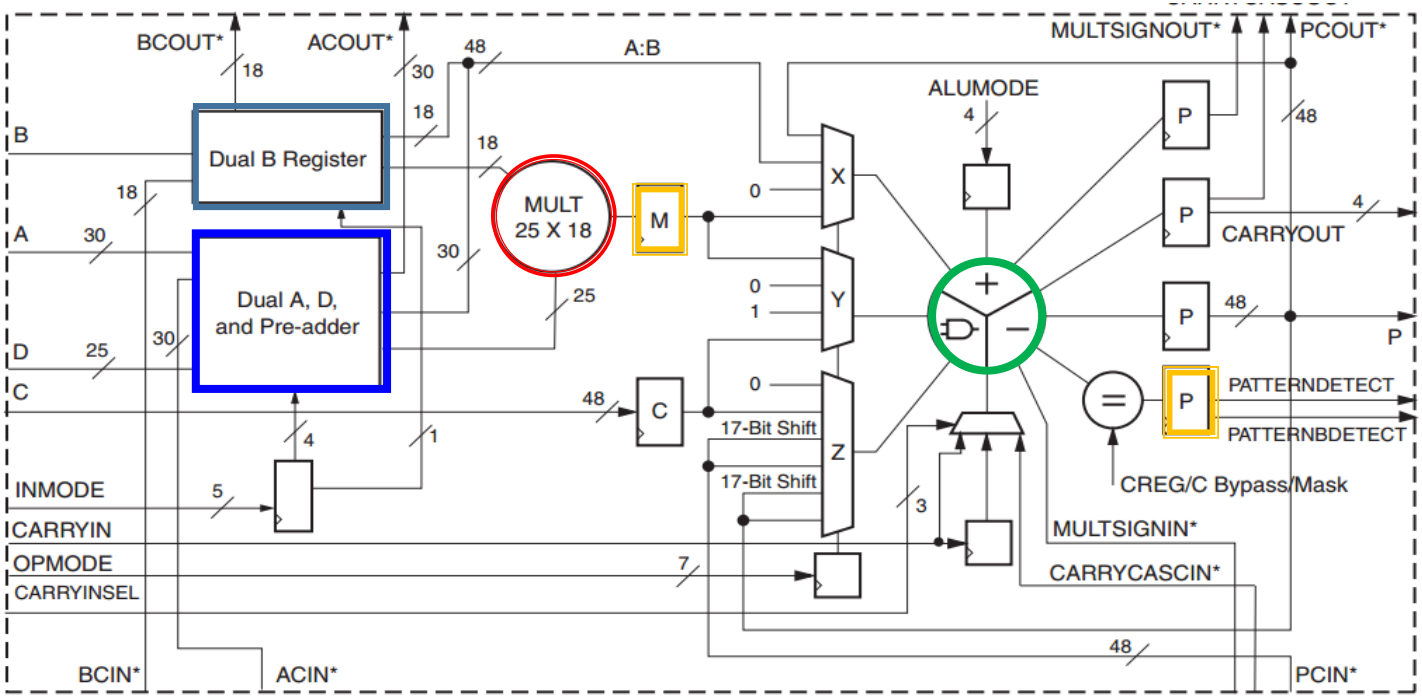
FIR Simétrico

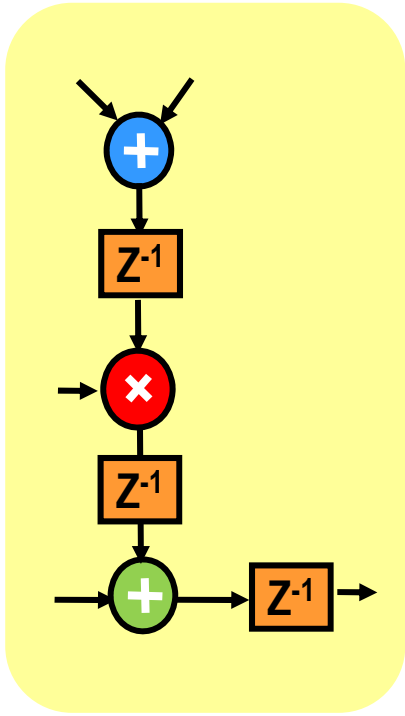


Xilinx
DSP48E

Segmentación en filtros FIR paralelos

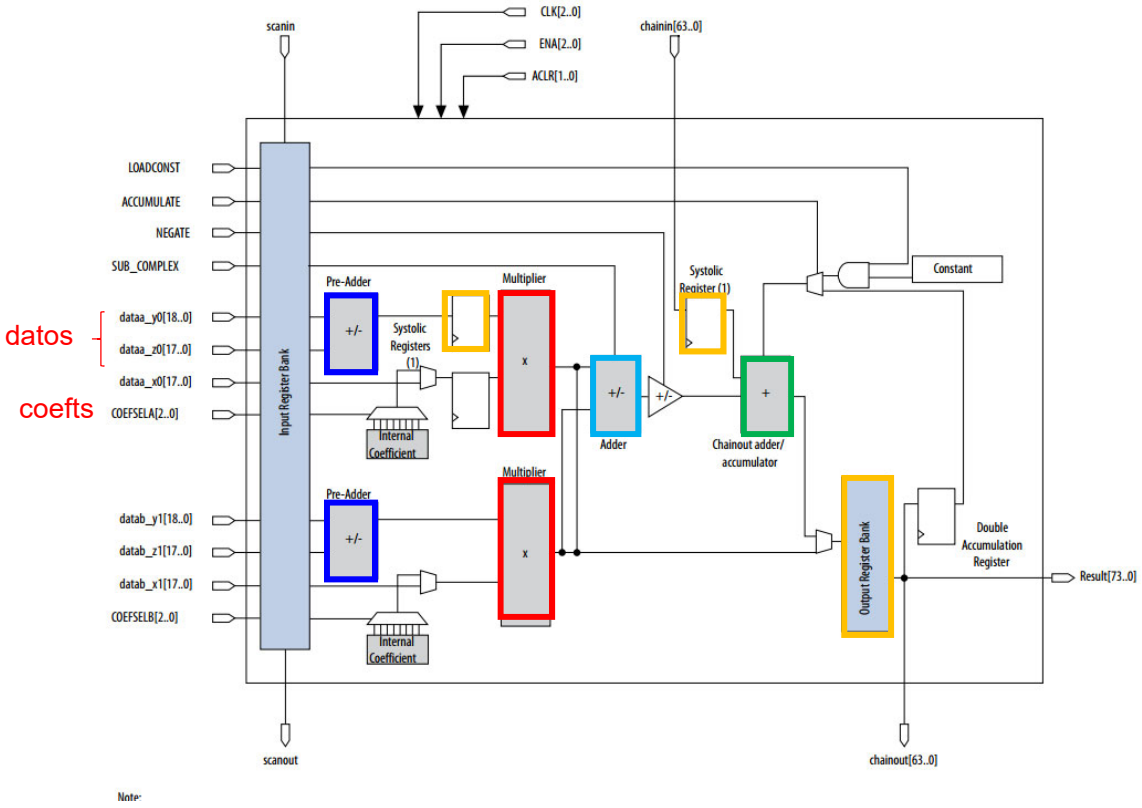
Xilinx DSP48E

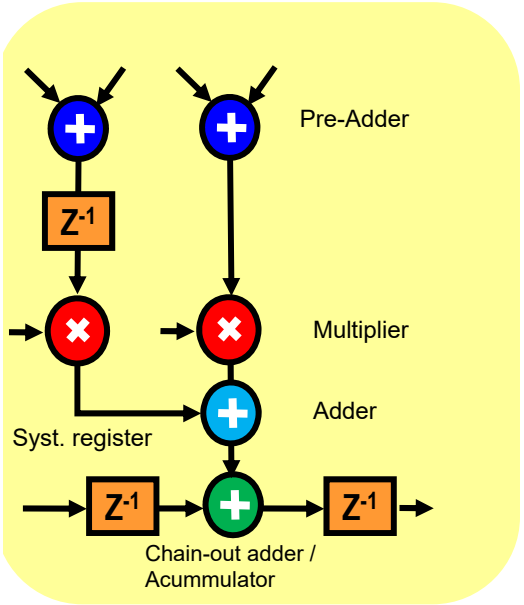




Segmentación en filtros FIR paralelos

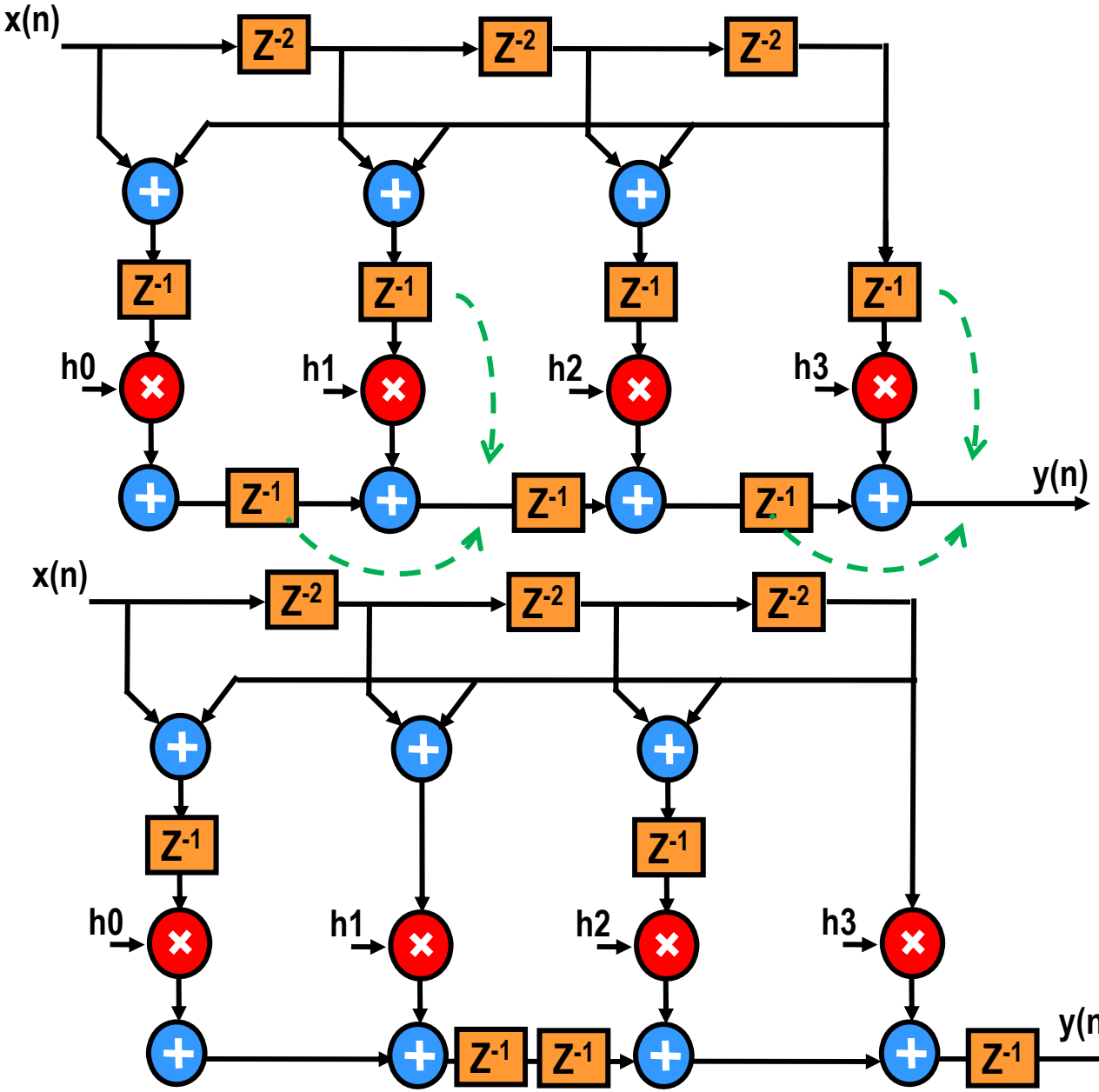
Altera _Cyclone V DSP Block
configurado como Filtro sistólico de 18 bits



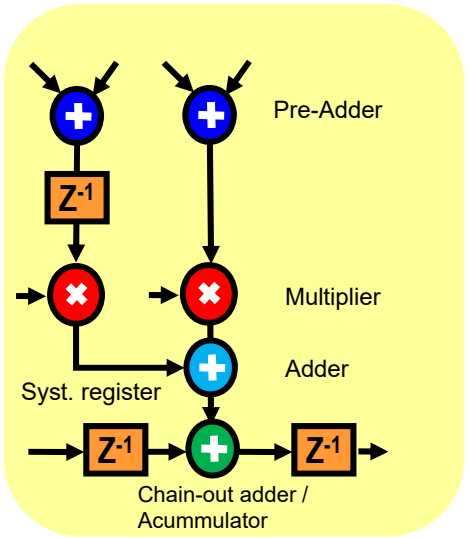


Permite almacenar 8 coeficientes constantes de 18 bits o de 27 bits

Segmentación en filtros FIR paralelos

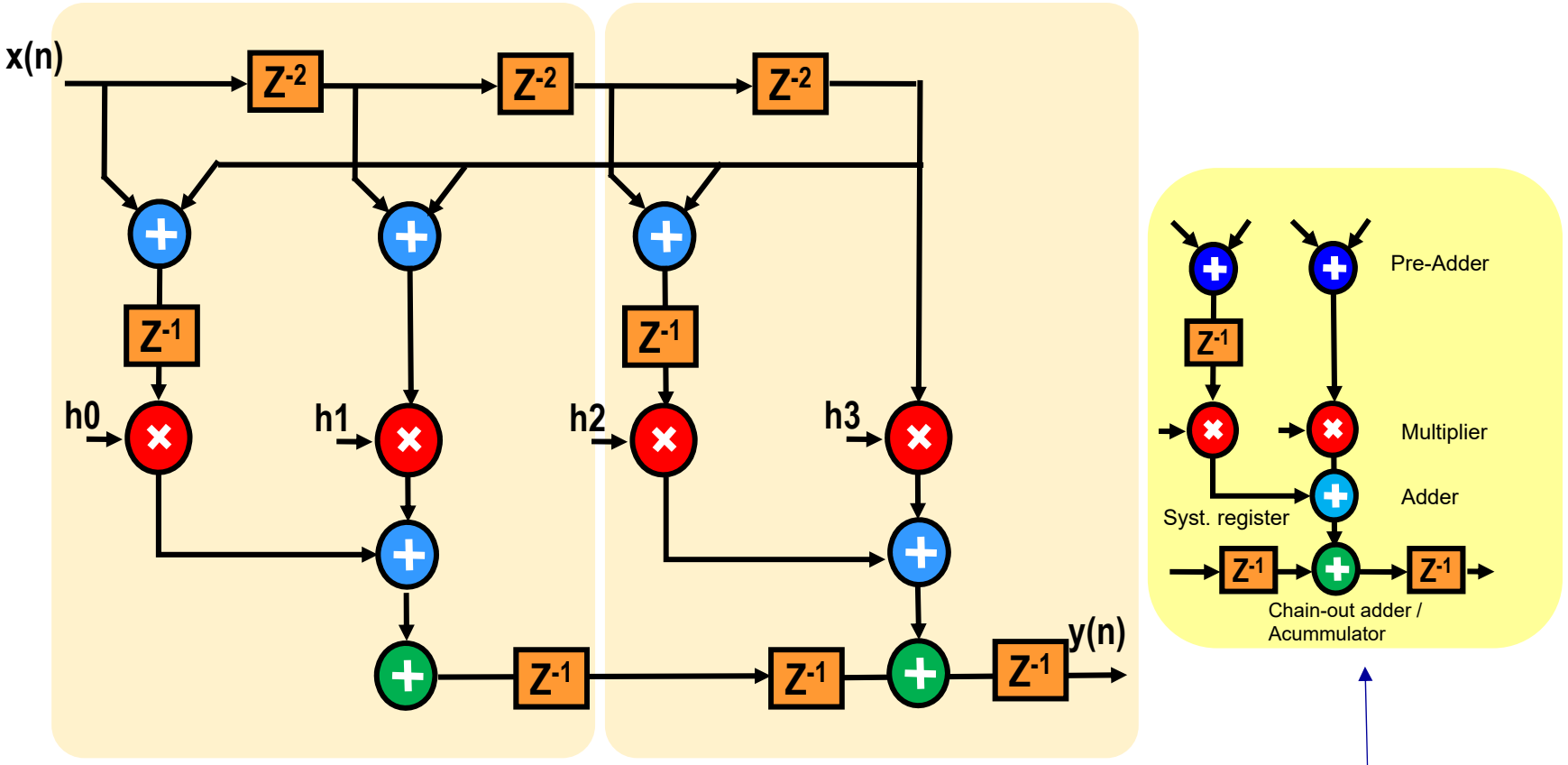


RETIMING



Segmentación en filtros FIR paralelos

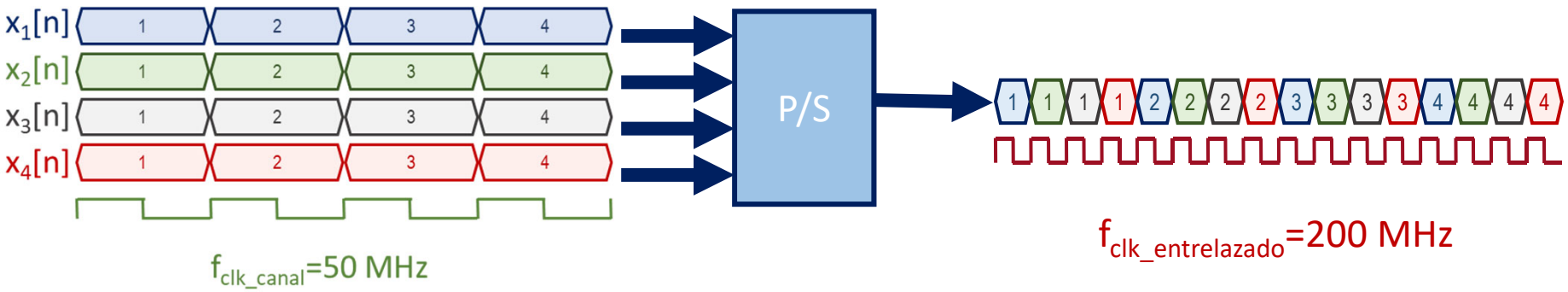
FIR Simétrico



Altera _Cyclone V DSP Block configurado como filtro sistólico de 18 bit

Procesado entrelazado

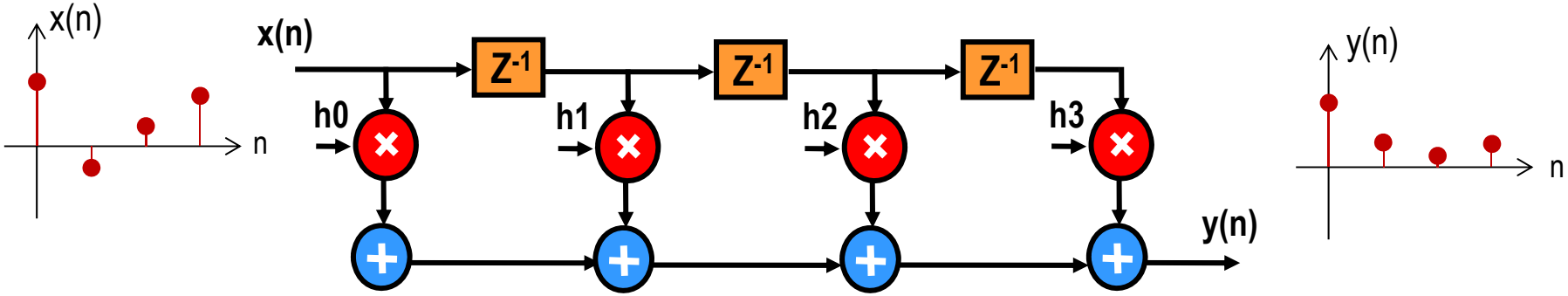
- Aplicaciones en las que se requiere aplicar el mismo procesado a múltiples flujos de datos independientes
  - Modulaciones digitales (ramas I y Q)
  - Interfaz de algunos conversores AD y DA



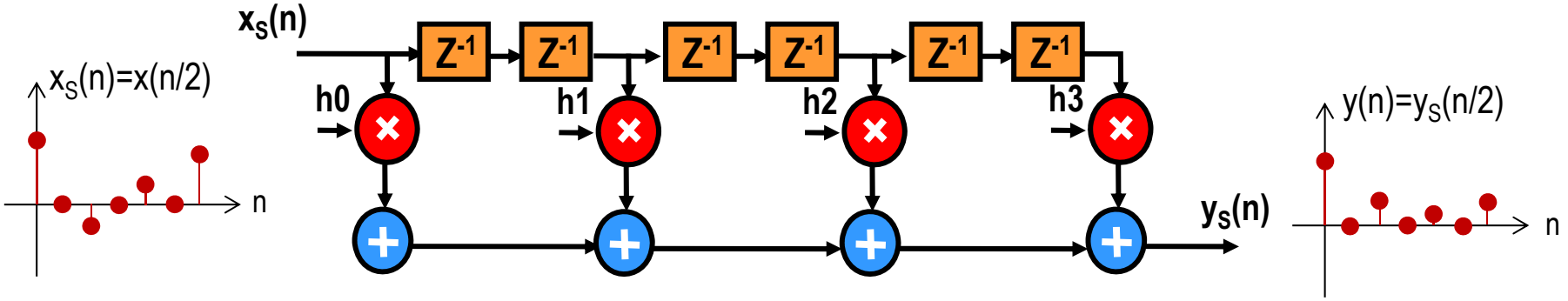
Procesado entrelazado

Escalado temporal

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3}$$

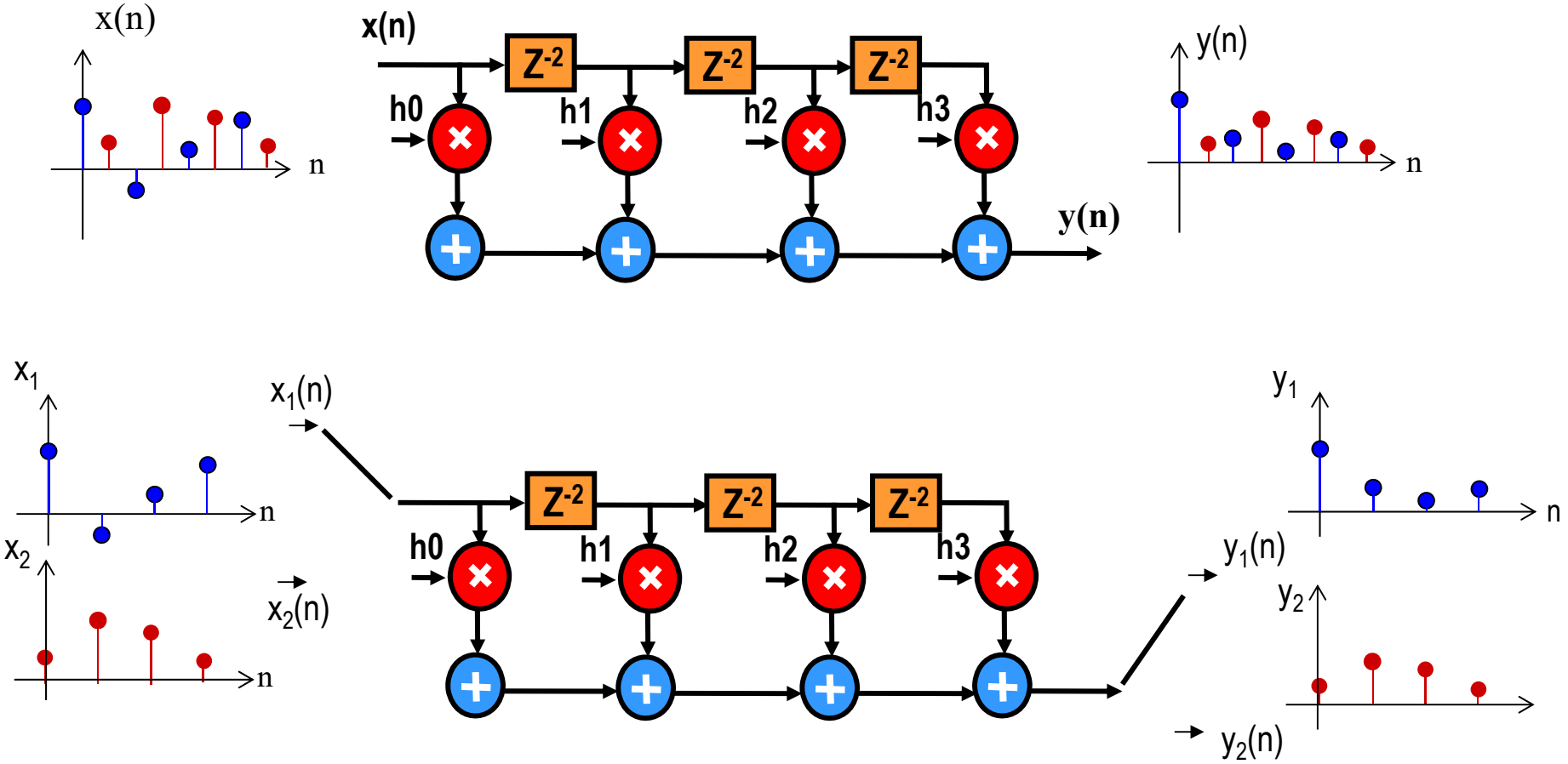


$$H_s(z) = H(z^2) \qquad H_s(z) = h_0 + h_1 z^{-2} + h_2 z^{-4} + h_3 z^{-6}$$



Procesado entrelazado

$$H_S(z) = H(z^2)$$



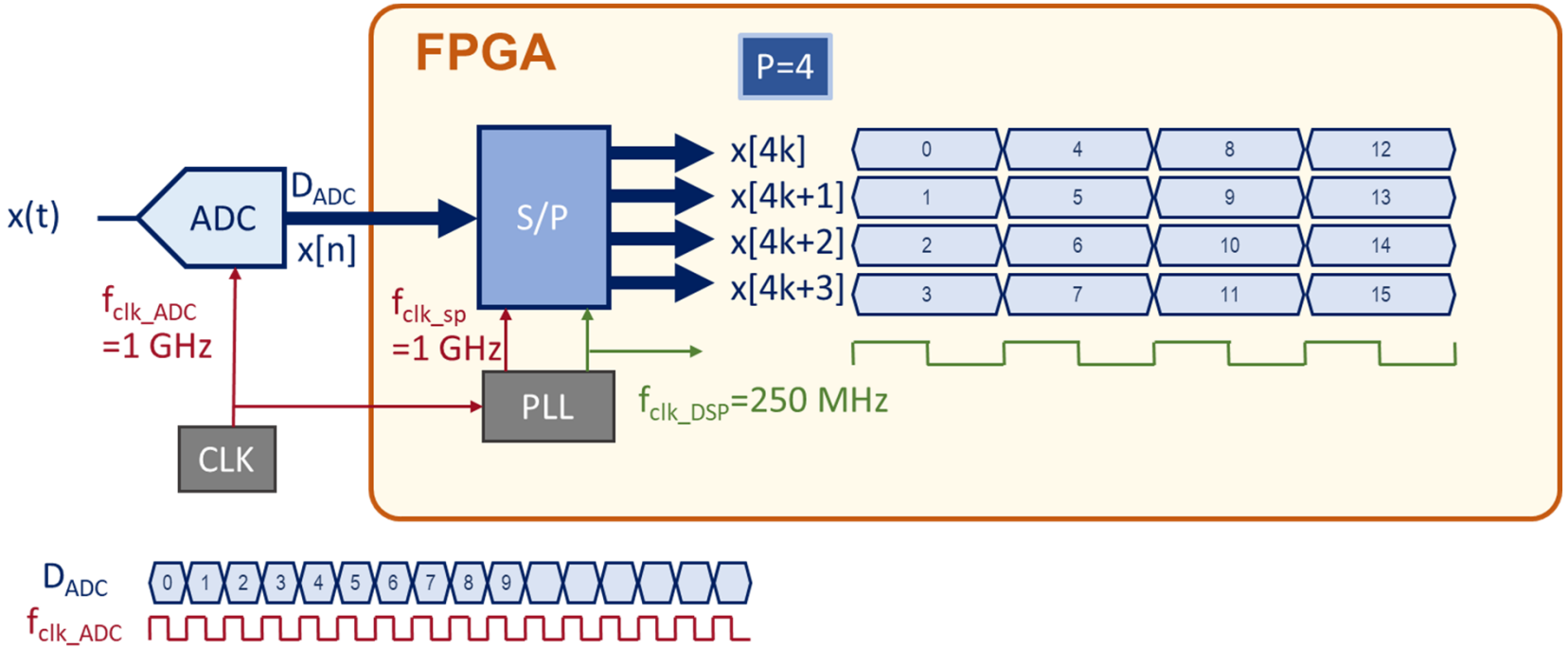
- $H(z^N) \rightarrow$ N flujos de datos se pueden procesar a $fs=f_{clk}/N$

Paralelización de algoritmos

- FPGA actuales → fclk_max< 500-600 MHz
  - ✓ en Cyclone IV fclk_max= 250 MHz
- Diseños grandes / FPGAs congestionada → difícil mantener una ruta de datos a fclk_max
- Aplicaciones que requieren el procesado de señales con BW grande → fs> fclk_max
  - Comunicaciones ópticas
  - Procesado digital de señales de RF
  - Microscopía con ultrasonidos
  - …
- Paralelización → el flujo de datos se divide en P flujos que se procesan a una fclk=fs/P
- Bloques IO de FPGAs → recursos para gestionar I/O a fclk altas y conversión a N flujos
  - ✓ Conversores P/S y S/P (SERDES: SERializer/DESserializer)
  - ✓ PLLs
  - ✓ Delays

Paralelización de algoritmos

- Ej: ADC 1 Gsps



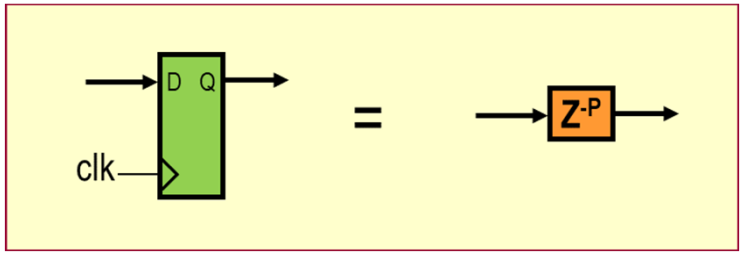
Paralelización de algoritmos

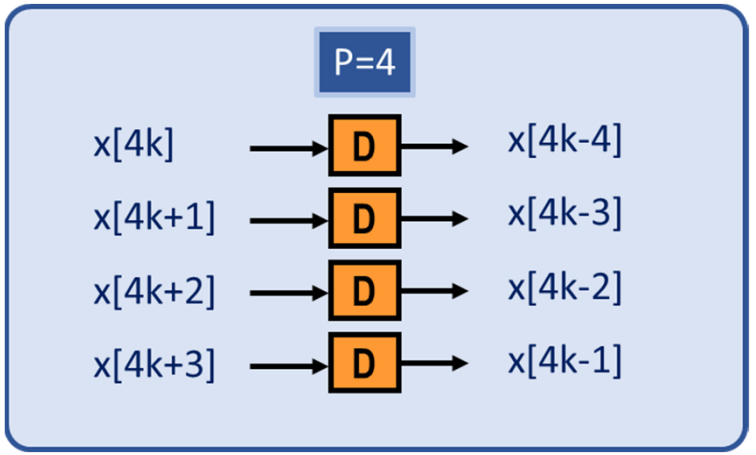
⇒Frecuencia de muestreo (fs) > Frecuencia de reloj (fclk) → fs=P·fclk

⇒La arquitectura necesita unas P veces los recursos que se requieren para implementar todas las operaciones del algoritmo

⇒Se obtienen P resultados en cada ciclo de reloj

⇒Un registro equivale a un retardo de P muestras temporales



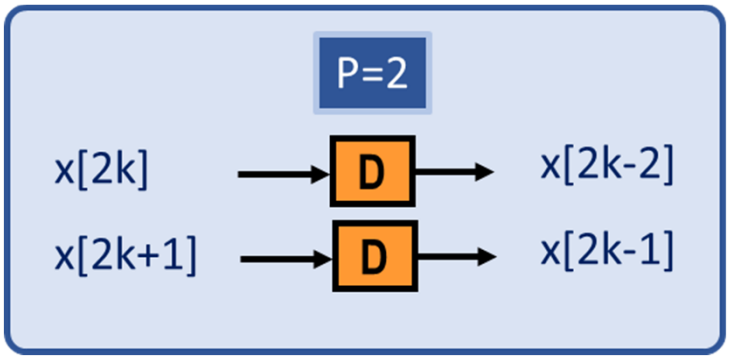


⇒**Métodos de diseño**:

- En el **tiempo** → ↑ complejidad de diseño eimplementación, ↓recursos
- En **z** mediante la **transf. polifásica** → ↓ complejidad, ↑recursos

Paralelización de algoritmos

Método de diseño en el tiempo:

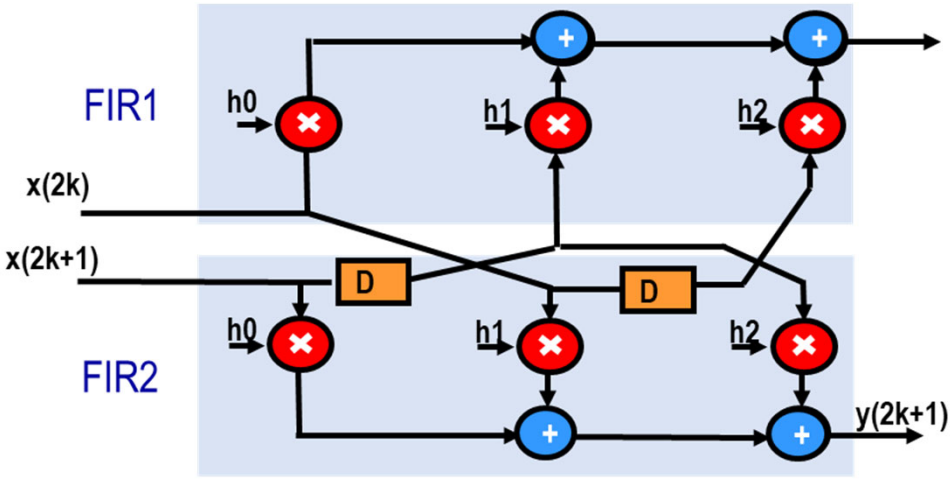


Ej. Filtro FIR de 3 etapas paralelizado por P=2

$$y(n) = h0 \cdot x(n) + h1 \cdot x(n-1) + h2 \cdot x(n-2)$$

P=2

$$y(2k) = h0 \cdot x(2k) + h1 \cdot x(2k-1) + h2 \cdot x(2k-2)$$
$$y(2k+1) = h0 \cdot x(2k+1) + h1 \cdot x(2k) + h2 \cdot x(2k-1)$$

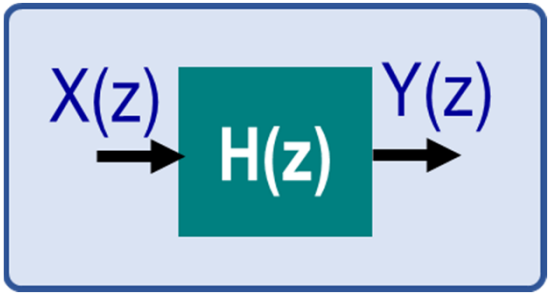


1) Evaluación y(n). en n=P·k, P·k+1, P·k+2,…,P·k+P-1
2) Implementación en paralelo cada ec.

Paralelización de algoritmos

Método de diseño basado en la descomposición polifásica:

1) Aplicación de la descom. en P fases a X(z), H(z) e Y(z)
2) Cálculo de Y(z)=H(z)·X(z)
3) Implementación en paralelo la ec. de cada fase de Y(z)



Ej. Filtro FIR de 3 etapas paralelizado por P=2

Descomp. polifásica en P=2 fases:

$$X(z)=x_0+x_1z^{-1}+x_2z^{-2}+x_3z^{-3}= (x_0+x_2z^{-2})+z^{-1}(x_1+x_3z^{-2})=X0(z^2)+z^{-1}X1(z^2)$$

1)
$$\begin{cases} X(z)=X0(z^2)+z^{-1}X1(z^2) \\ H(z)=H0(z^2)+z^{-1}H1(z^2) \\ Y(z)=Y0(z^2)+z^{-1}Y1(z^2) \end{cases}$$

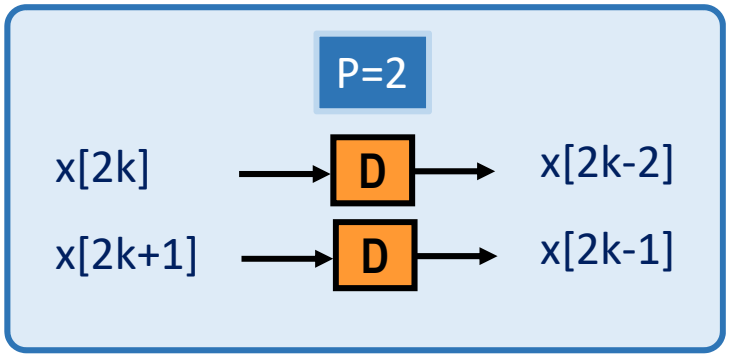
2)
$$\begin{cases} Y(z)=H(z)X(z) = [X0(z^2)+z^{-1}X1(z^2)]\cdot[H0(z^2)+z^{-1}H1(z^2)]= \\ =X0(z^2)H0(z^2)+z^{-1}X0(z^2)H1(z^2) +z^{-1}X1(z^2)H0(z^2)+z^{-2}X1(z^2)H1(z^2)= \\ =Y0(z^2)+z^{-1}Y1(z^2) \end{cases}$$

$$\begin{cases} Y0(z^2) = X0(z^2)H0(z^2) +z^{-2}X1(z^2)\ H1(z^2) \\ Y1(z^2) = X0(z^2)H1(z^2) +X1(z^2)H0(z^2) \end{cases}$$

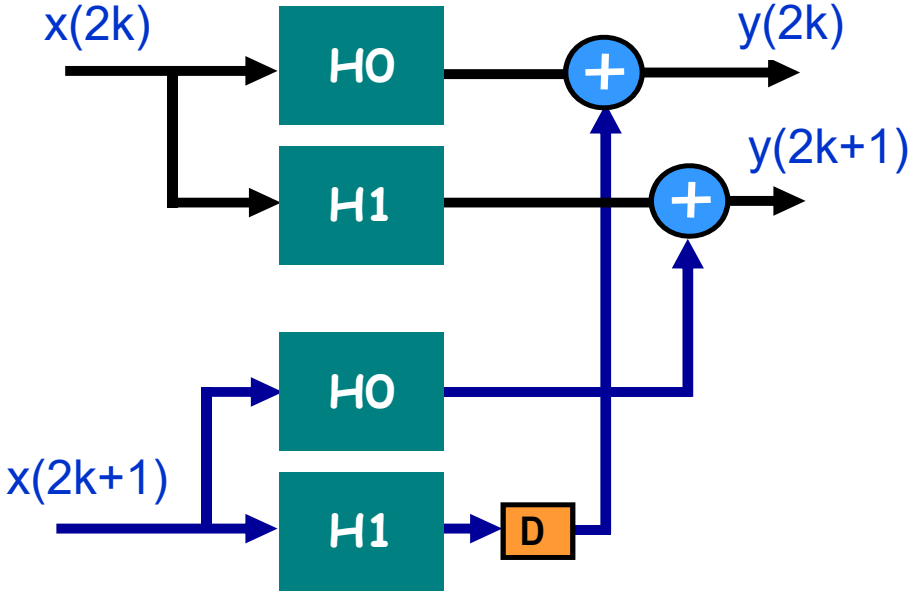
Paralelización de algoritmos

Ej. Filtro FIR de 3 etapas paralelizado por P=2

2) $$\begin{cases} Y0(z^2) = X0(z^2)H0(z^2) + z^{-2}X1(z^2)H1(z^2) \\ Y1(z^2) = X0(z^2)H1(z^2) + X1(z^2)H0(z^2) \end{cases}$$



3)

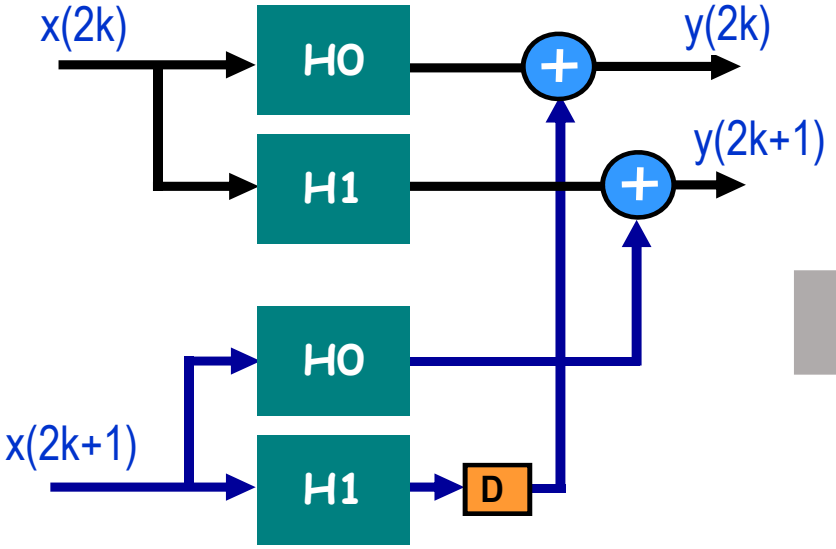


Paralelización de algoritmos

Ej. Filtro FIR de 3 etapas paralelizado por P=2

El área de un filtro FIR paralelo puede reducirse explotando las características de la estructura

$$\begin{cases} Y0 = X0H0 + z^{-2}X1H1 \\ Y1 = X0H1 + X1H0 \end{cases}$$



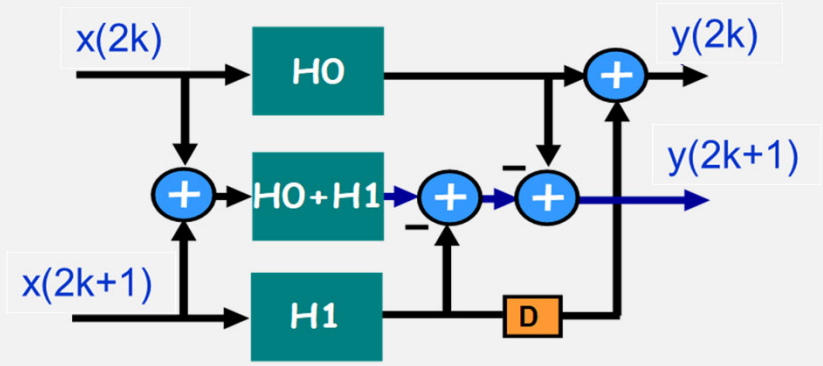
Reducción de la complejidad

$Y0 = X0H0 + z^{-2}X1H1$

$Y1 = (X0+X1)(H0+H1) - X0H0 - X1H1$

Nuevo bloque coef: H0+H1

Bloques compartidos



Conclusiones

- Estudio, diseño e implementación de Arquitecturas Paralelas
- Técnicas para mejorar las prestaciones de los circuitos y adaptar sus estructuras para implementarlas en distintos dispositivos FPGA
- Uso eficiente de los recursos DSP de las FPGA de Xilinx y Altera
- Técnicas de Paralelización de algoritmos

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Arquitecturas Paralelas

Procesado Digital de la Señal en FPGA

Máster Universitario en Ingeniería de Sistemas Electrónicos