

Diseño Digital Avanzado

Unidad 2 - Modelo de Imp. de Sistemas

Dr. Ariel L. Pola

apola@fundacionfulgor.org.ar
September 1, 2022

Tabla de Contenidos

1. Contenidos Temáticos
2. Introducción
3. Estructuras para sistemas FIR
4. Estructuras para sistemas IIR
5. Representaciones Numéricas
6. Formato de Punto Flotante
7. Aritmética de Punto Fijo

Contenidos Temáticos



Presentación del Curso

Contenidos Temáticos

Unidad 2 Modelo de Implementación de Sistemas

- Introducción
- Comparación entre arquitecturas de punto fijo y punto flotante.
- Repaso representaciones numéricas.
- Representación en complemento a 2.
- Formatos de punto flotante y punto fijo.
- Conversión de números de punto flotante a punto fijo.
- Operaciones en ambos sistemas.
- Saturación y Overflow.
- Redondeo y truncamiento.
- Soporte de Python para punto fijo.
- Formas de Filtros Digitales.
- Cuantización de los coeficientes de filtros IIR y FIR.
- Generación de vectores para verificación de código RTL.

Introducción



Estructuras de Filtros Digitales

Introducción

- Existen varias configuraciones o estructuras para la realización de cualquier sistema discreto en el tiempo FIR e IIR.
- Las estructuras pueden ser
 - Forma Directa I
 - Forma Directa II
 - Cascada
 - Paralela
- Presentan robustez frente a los efectos de cuantización.

Estructuras de Filtros Digitales

Introducción

- Consideraremos los sistemas discretos en el tiempo lineales e invariantes en el tiempo caracterizada por la ecuación en diferencias general y lineal de coeficientes constantes

$$y(n) = - \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (1)$$

donde utilizando la transformada Z, los sistemas se caracterizan por la **función de transferencia**

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (2)$$

- El objetivo que buscamos es la mejor forma de implementar el filtro reduciendo
 - La complejidad del cálculo (multiplicaciones y sumas)
 - Los requisitos de memoria
 - Los efectos de la longitud de palabra finita sobre los cálculos
- Implementación en punto fijo o punto flotante
- Incluyen paralelización o pipeline.

Estructuras para sistemas FIR



Estructuras de Filtros Digitales

Estructuras para sistemas FIR

- En general, un sistema FIR se describe mediante la ecuación en diferencias

$$y(n) = \sum_{k=0}^{M-1} b_k x(n-k) \quad (3)$$

o, lo que es equivalente, mediante la función del sistema

$$H(z) = \sum_{k=0}^{M-1} b_k z^{-k} \quad (4)$$

Además, la respuesta al impulso unitario del sistema FIR es idéntica a los coeficientes b_k , es decir,

$$h(n) = \begin{cases} b_n & 0 < n < M-1 \\ 0 & \text{en otro caso} \end{cases} \quad (5)$$

donde M es la longitud del filtro FIR.

Estructuras de Filtros Digitales

Estructuras para sistemas FIR

Forma Directa

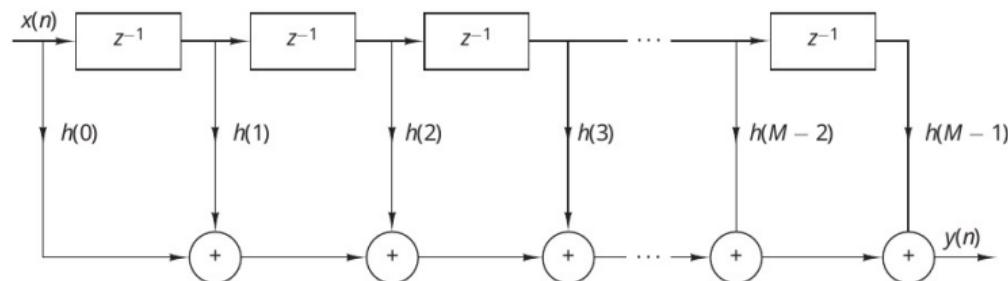
- Se obtiene de la ecuación en diferencias no recursiva

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (6)$$

- Características

- $M - 1$ posiciones de memoria
- M multiplicaciones
- $M - 1$ sumas

- Se los conocen como filtros Transversales



Estructuras de Filtros Digitales

Estructuras para sistemas FIR

Forma Directa

- Cuando un sistema FIR tiene una **fase lineal**, la respuesta al impulso unitario del sistema satisface la condición de simetría o la condición de antisimetría

$$h(n) = \pm h(M - 1 - n) \quad (7)$$

- El número de multiplicaciones se reduce de M a
 - $M/2$ para M par
 - $(M - 1)/2$ para M impar

Estructuras de Filtros Digitales

Filtro FIR Plegado (*Folded FIR Filter*)

- Desde la perspectiva del hardware, un multiplicador toma más área que un sumador.
- El diseño del filtro FIR puede ser simétrico o anti-simétrico y matemáticamente se representa como

$$h[M-n] = \pm h[n], \quad n = 0, 1, 2, \dots, M \quad (8)$$

- Esta característica del filtro FIR puede utilizarse eficazmente para reducir el número de multiplicadores en la implementación de hardware.
- Por ejemplo, si el filtro tiene cuatro coeficientes simétricos, la suma de convolución se escribe como:

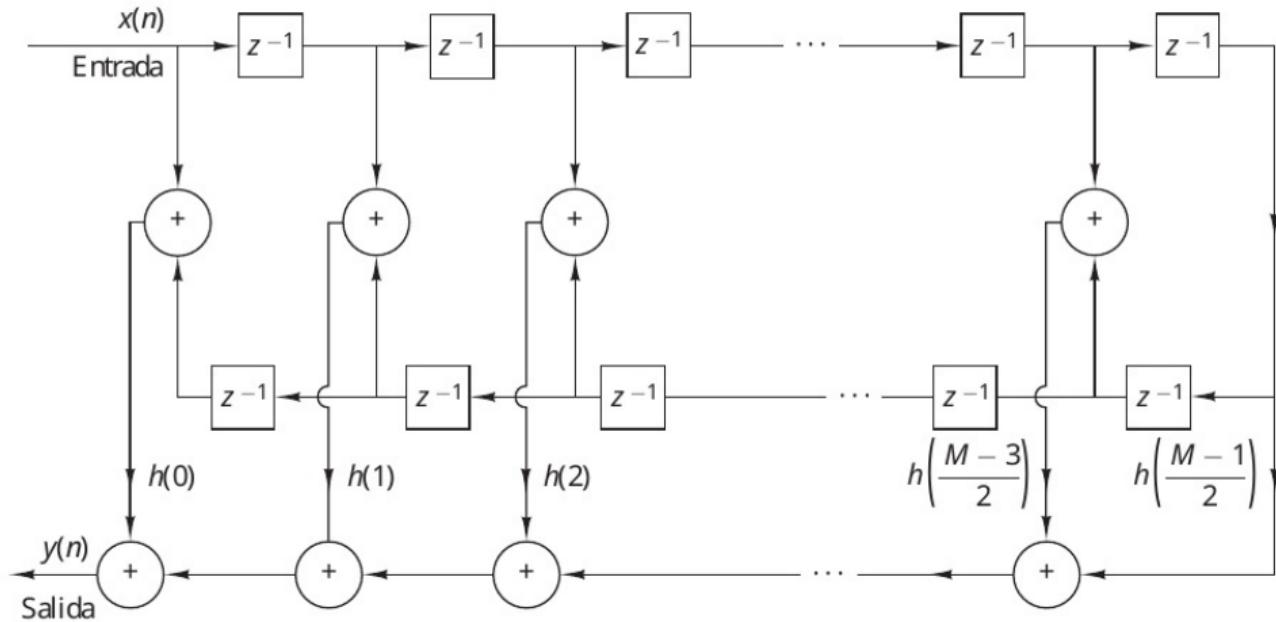
$$y[n] = h_0x_n + h_1x_{n-1} + h_1x_{n-2} + h_0x_{n-3} \quad (9)$$

- Finalmente agrupando términos observamos que solo se realizan dos operaciones de multiplicación.

$$y[n] = h_0(x_n + x_{n-3}) + h_1(x_{n-1} + x_{n-2}) \quad (10)$$

Estructuras de Filtros Digitales

Estructuras para sistemas FIR



Realización en la forma directa de un sistema FIR de fase lineal (M impar).

Estructuras de Filtros Digitales

Estructuras para sistemas FIR

Cascada

- Factorizando $H(z)$ en los sistemas FIR de segundo orden de modo que

$$H(z) = \prod_{k=1}^K H_k(z) \quad (11)$$

donde

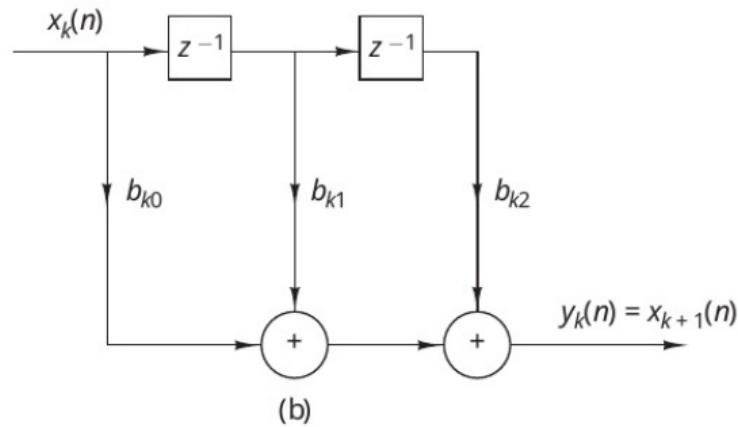
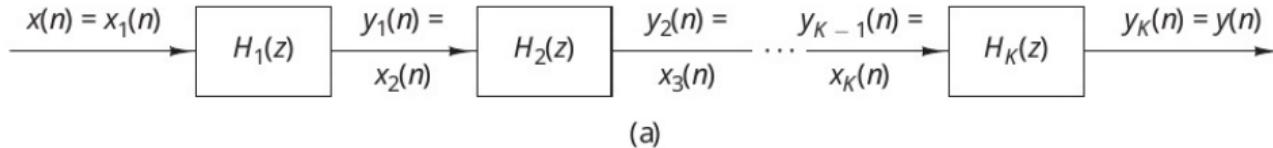
$$H_k(Z) = b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}, \quad k = 1, 2, \dots, K \quad (12)$$

y K es la parte entera de $(M+1)/2$.

- El parámetro del filtro b_0 puede distribuirse igualmente entre las K secciones del filtro, tal que $b_0 = b_{10}b_{20}\dots b_{K0}$ o puede asignarse a una sola sección del filtro.
- Los ceros de $H(Z)$ se agrupan por parejas para generar sistemas FIR de segundo orden.
- Las parejas de raíces complejas conjugadas hacen que los coeficientes b_{ki} sean valores reales.

Estructuras de Filtros Digitales

Estructuras para sistemas FIR



Realización en cascada de un sistema FIR

Estructuras de Filtros Digitales

Estructuras basadas en el muestreo en frecuencia

- Los parámetros que caracterizan el filtro son los valores de la respuesta en frecuencia.
- Se define la respuesta en frecuencia deseada en un conjunto de frecuencias igualmente espaciadas

$$\begin{aligned}\omega_k &= \frac{2\pi}{M}(k + \alpha), \quad k = 0, 1, \dots, \frac{M-1}{2}, \quad M \text{ impar} \\ k &= 0, 1, \dots, \frac{M}{2} - 1, \quad M \text{ par} \\ \alpha &= 0 \text{ o } \frac{1}{2}\end{aligned}$$

- La respuesta en frecuencia se puede escribir como

$$H(\omega) = \sum_{n=0}^{M-1} h(n) e^{-j\omega n} \tag{13}$$

- Los valores de $H(\omega)$ en las frecuencias $\omega_k = (2\pi/M)(k + D)$ son

$$H(k + \alpha) = H\left(\frac{2\pi}{M}(k + \alpha)\right) = \sum_{n=0}^{M-1} h(n) e^{-j2\pi(k+\alpha)n/M}, \quad k = 0, 1, \dots, M-1 \tag{14}$$

- En el caso en que $\alpha = 0$, $H(k)$ se corresponde con la DFT de M puntos de $h(n)$.

Estructuras de Filtros Digitales

Estructuras basadas en el muestreo en frecuencia

- Si expresamos $h(n)$ en función de las muestras en frecuencia

$$h(n) = \frac{1}{M} \sum_{k=0}^{M-1} H(k + \alpha) e^{j2\pi(k+\alpha)n/M}, \quad n = 0, 1, \dots, M-1 \quad (15)$$

- Si $\alpha = 0$ obtenemos la IDFT de $H(k)$.
- Si sustituimos $h(n)$ en la expresión de la transformada z y reagrupamos

$$H(z) = \frac{1 - z^{-M} e^{j2\pi\alpha}}{M} \sum_{k=0}^{M-1} \frac{H(k + \alpha)}{1 - e^{j2\pi(k+\alpha)/M} z^{-1}} \quad (16)$$

- Por tanto, la función del sistema $H(z)$ queda caracterizada por el conjunto de muestras en frecuencia $H(k + D)$ en lugar de por $h(n)$.

Estructuras de Filtros Digitales

Estructuras basadas en el muestreo en frecuencia

- Filtro FIR de dos filtros en cascada $H(z) = H_1(z)H_2(z)$.

- Filtro de todo ceros, o filtro peine

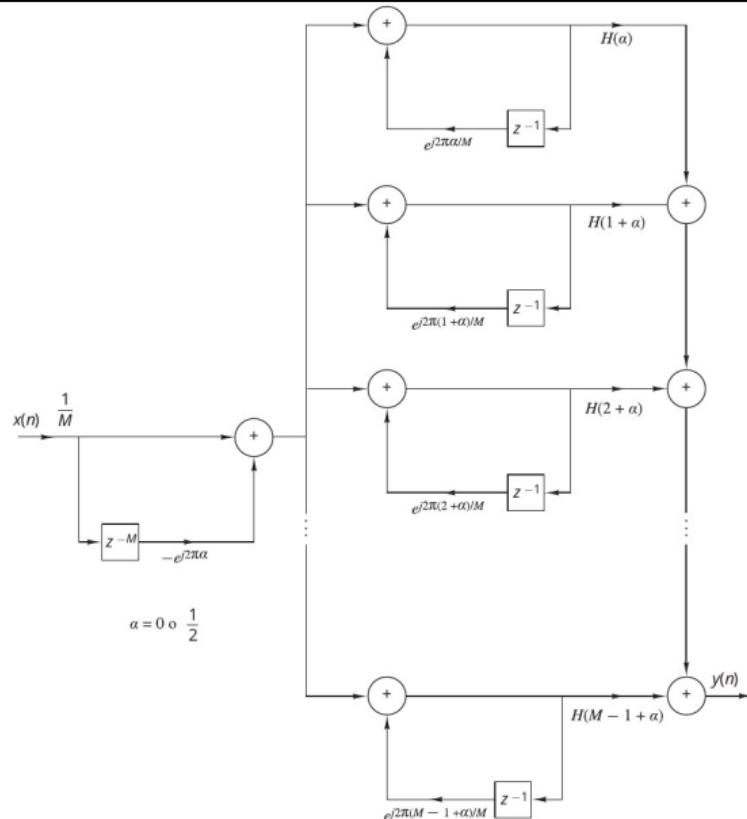
$$H_1(z) = \frac{1 - z^{-M} e^{j2\pi\alpha}}{M} \quad (17)$$

- El segundo filtro consta de un banco paralelo de filtros de un solo polo

$$H_2(z) = \sum_{k=0}^{M-1} \frac{H(k + \alpha)}{1 - e^{j2\pi(k+\alpha)/M} z^{-1}} \quad (18)$$

Estructuras de Filtros Digitales

Estructuras basadas en el muestreo en frecuencia



Estructuras de Filtros Digitales

Cuantización de Coeficientes

- Como los sistemas FIR son siempre estables, la estabilidad no es una preocupación al cuantificar los coeficientes.
- La cuantización del coeficiente añade principalmente una respuesta de frecuencia adicional indeseable a la respuesta de frecuencia del sistema original.
- El razonamiento matemático de esta adición como

$$h_Q[n] = h[n] + \Delta h[n] \quad (19)$$

$$H_Q(e^{j\omega}) = \sum_{n=0}^M (h[n] + \Delta h[n]) e^{j\omega} \quad (20)$$

$$H_Q(e^{j\omega}) = H(e^{j\omega}) + \sum_{n=0}^M \Delta h[n] e^{j\omega} \quad (21)$$

- Así, la cuantización de coeficientes de un filtro FIR añade una respuesta de frecuencia igual a la respuesta en frecuencia de $\Delta h[n]$ causada por la cuantización de $h[n]$.

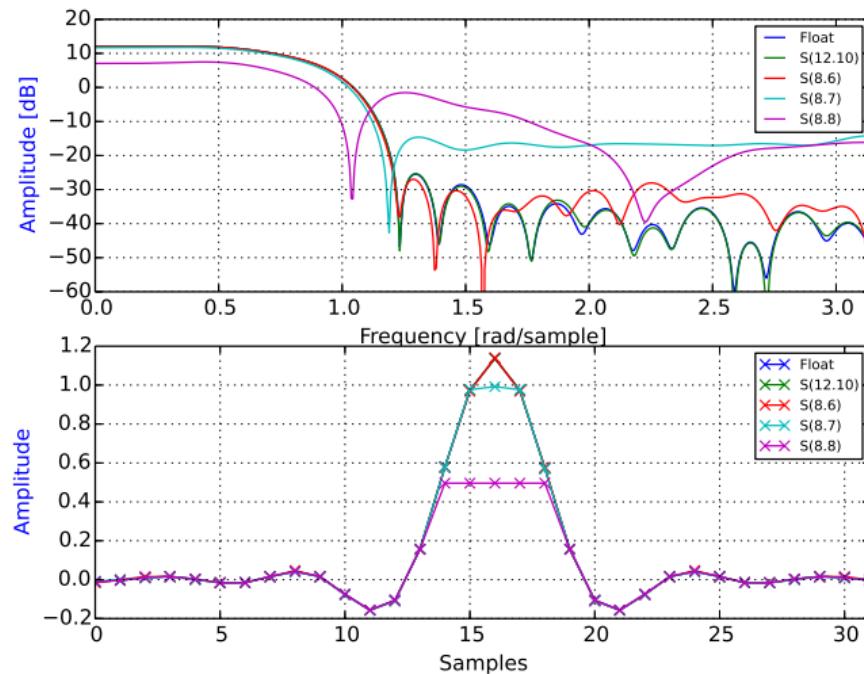
Estructuras de Filtros Digitales

Cuantización de Coeficientes

■ Estrategia de diseño

- La cuantización de coeficientes obviamente cambia la respuesta de frecuencia del filtro.
- Existe la posibilidad de que el filtro cuantizado ya no pueda satisfacer las especificaciones de diseño originales.
- El diseñador puede necesitar modificar el filtro para permitir que los efectos de cuantizados cumplan con las especificaciones.
- Una solución eficiente para evitar el re-diseño del filtro es realizarlo directamente en punto fijo.
- Muy pocos diseñadores consideran como punto de partida la elaboración del filtro en punto fijo.
- Una de las causas es la alta carga computacional que implica utilizar clases de punto fijo.

Estructuras de Filtros Digitales



Efecto de la cuantización sobre filtros FIR.

Estructuras para sistemas IIR



Estructuras de Filtros Digitales

Estructuras para sistemas IIR

- Las diferentes estructuras de los sistemas IIR se pueden describir por la siguiente ecuación en diferencias y función de sistema

$$y(n) = - \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (22)$$

$$H(z) = \frac{\sum_{k=0}^M b_k z^{-k}}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (23)$$

- Tipos de estructuras:

- Forma directa I y II
- Cascada
- Paralela

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Forma Directa I

- La función de sistema racional que caracteriza un sistema IIR puede interpretarse como dos sistemas conectados en cascada, es decir,

$$H(z) = H_1(z)H_2(z) \quad (24)$$

donde $H_1(z)$ consta de los ceros de $H(z)$ y $H_2(z)$ consta de los polos de $H(z)$,

$$H_1(z) = \sum_{k=0}^M b_k z^{-k} \quad (25)$$

y

$$H_2(z) = \frac{1}{1 + \sum_{k=1}^N a_k z^{-k}} \quad (26)$$

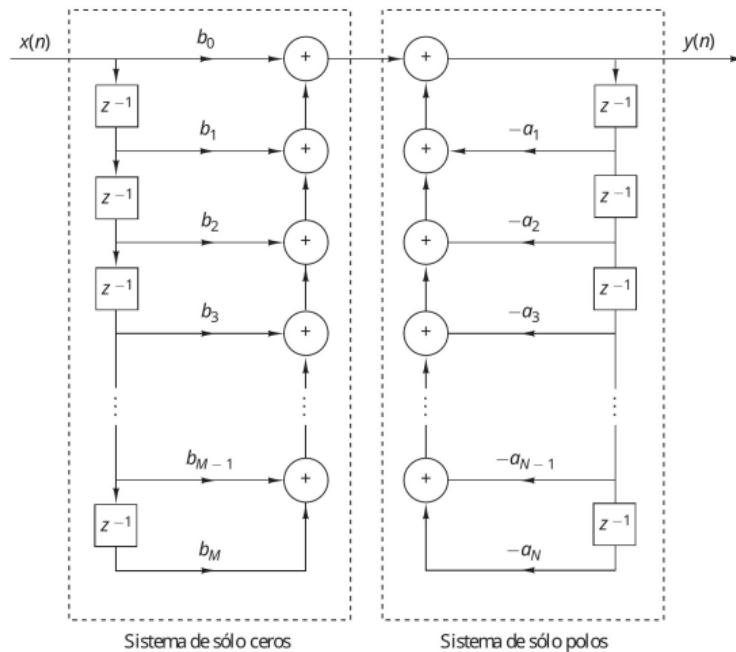
- Dependiendo el orden de los filtros $H_1(z)$ y $H_2(z)$ podemos obtener las estructuras Formas Directa I y II.

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Forma Directa I

- Requiere $M + N + 1$ multiplicaciones, $M + N$ sumas y $M + N + 1$ posiciones de memoria.



Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Forma Directa II

- Si el filtro de sólo polos $H_2(z)$ se coloca antes que el filtro de sólo ceros $H_1(z)$, se consigue una estructura más compacta.
- La ecuación en diferencias del filtro de sólo polos es

$$w(n) = - \sum_{k=1}^N a_k w(n-k) x(n) \quad (27)$$

- Dado que $w(n)$ es la entrada al sistema de sólo ceros, su salida es

$$y(n) = \sum_{k=0}^M b_k w(n-k) \quad (28)$$

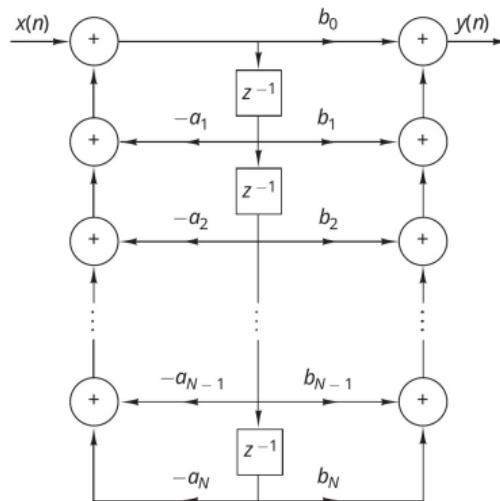
- Observe que en ambas expresiones $w(n)$ es una secuencia de retardos.

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Forma Directa II

- Requiere $M + N + 1$ multiplicaciones, $M + N$ sumas y un máximo de M, N posiciones de memoria.
- Ambas estructuras son extremadamente sensibles a los parámetros de cuantización, por lo que no se recomienda su uso.

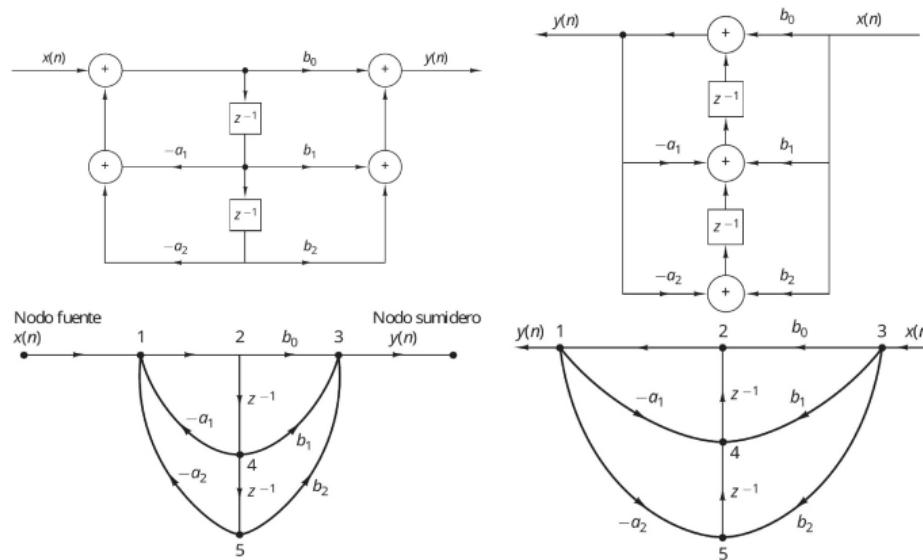


Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructuras Transpuestas

- **Teorema de transposición o teorema del diagrama de flujo inverso:** Si invertimos las direcciones de todas las transmitancias de rama e intercambiamos la entrada y la salida en el diagrama de flujo, la función del sistema no varía.



Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructuras Transpuestas

- La realización transpuesta de la forma directa II se expresa como

$$y(n) = w_1(n-1) + b_0x(n) \quad (29)$$

$$w_k(n) = w_{k+1}(n-1) - a_ky(n) + b_kx(n) \quad k = 1, 2, \dots, N-1 \quad (30)$$

$$w_N(n) = b_Nx(n) - a_Ny(n) \quad (31)$$

- Generalizando,

$$y(n) = - \sum_{k=1}^N a_k y(n-k) + \sum_{k=0}^M b_k x(n-k) \quad (32)$$

- La estructura transpuesta en la forma II requiere el mismo número de multiplicaciones, sumas y posiciones de memoria que la estructura original en la forma directa II.

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructura	Ecuaciones de implementación	Función del sistema
<p>Forma directa I</p>	$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2)$ $- a_1y(n-1) - a_2y(n-2)$	$\frac{H(z) = b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$
<p>Forma directa regular II</p>	$w(n) = -a_1w(n-1) - a_2w(n-2) + x(n)$ $y(n) = b_0w(n) + b_1w(n-1) + b_2w(n-2)$	$\frac{H(z) = b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$
<p>Forma directa transuesta II</p>	$y(n) = b_0x(n) + w_1(n-1)$ $w_1(n) = b_1x(n) - a_1y(n) + w_2(n-1)$ $w_2(n) = b_2x(n) - a_2y(n)$	$\frac{H(z) = b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}}$

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructuras en Cascada

- El sistema puede descomponerse como

$$H(z) = \prod_{k=1}^K H_k(z) \quad (33)$$

donde

$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1} + b_{k2}z^{-2}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}}, \quad k = 1, 2, \dots, K \quad (34)$$

y K es la parte entera de $(M+1)/2$.

- El parámetro del filtro b_0 puede distribuirse igualmente entre las K secciones del filtro, tal que $b_0 = b_{10}b_{20}\dots b_{K0}$.
- Los coeficientes a_{ki} y b_{ki} de los sub-sistemas de segundo orden son reales.
- Esto implica que al configurar los sub-sistemas de segundo orden se agrupará una pareja de polos complejos conjugados y una pareja de ceros complejos conjugados.

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructuras en Cascada

- Cada uno de los sub-sistemas de segundo orden puede implementarse según la forma directa I, la forma directa II o la forma directa transpuesta II.
- La función $H(z)$ es

$$y_0(n) = x(n)$$

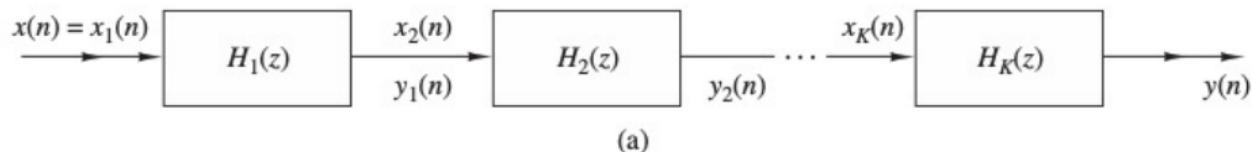
$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + y_{k-1}(n), \quad k = 1, 2, \dots, K$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1) + b_{k2}w_k(n-2), \quad k = 1, 2, \dots, K$$

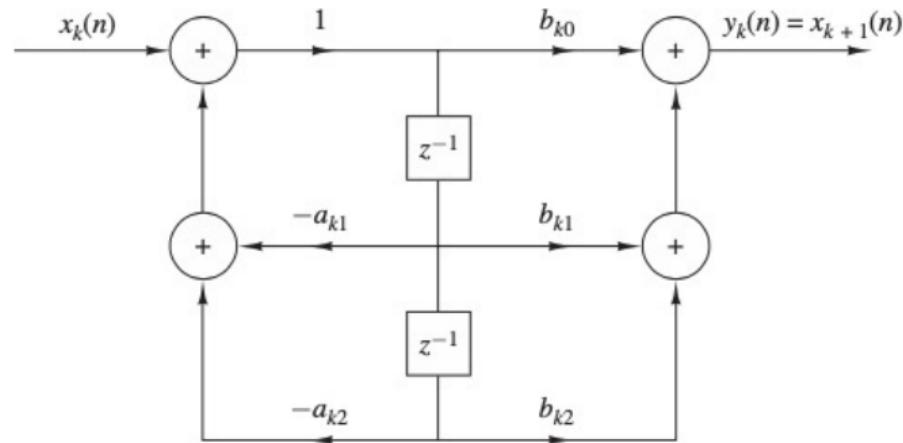
$$y(n) = y_K(n)$$

Estructuras de Filtros Digitales

Estructuras en forma directa IIR



(a)



Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Estructuras en paralelo

- Expandiendo en fracciones parciales de $H(z)$ y suponiendo que $N \geq M$ y polos distintos obtenemos

$$H(z) = C + \sum_{k=1}^N \frac{A_k}{1 - p_k z^{-1}} \quad (35)$$

donde p_k son los polos, A_k son los coeficientes de la expansión en fracciones parciales y $C = b_N/a_N$.

- Cada sub-sistema es

$$H_k(z) = \frac{b_{k0} + b_{k1}z^{-1}}{1 + a_{k1}z^{-1} + a_{k2}z^{-2}} \quad (36)$$

donde b_{ki} y a_{ki} son los parámetros reales del sistema.

Estructuras de Filtros Digitales

Estructuras en forma directa IIR

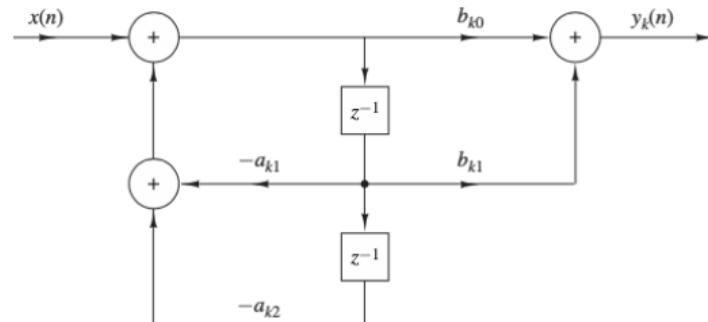
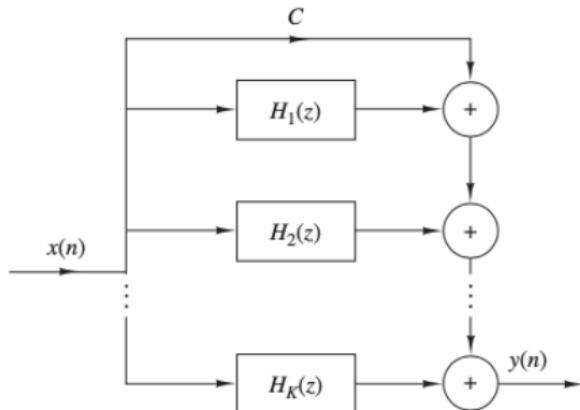
Estructuras en paralelo

- La representación paralela para forma directa II es

$$w_k(n) = -a_{k1}w_k(n-1) - a_{k2}w_k(n-2) + x(n), \quad k = 1, 2, \dots, K$$

$$y_k(n) = b_{k0}w_k(n) + b_{k1}w_k(n-1), \quad k = 1, 2, \dots, K$$

$$y(n) = Cx(n) + \sum_{k=1}^K y_k(n)$$



Estructuras de Filtros Digitales

Estructuras en forma directa IIR

Implementaciones

- Aunque todas estas implementaciones son analíticamente equivalentes, varían en términos de requerimientos de recursos de hardware y su sensibilidad a la cuantización de coeficientes.
- Al implementar un algoritmo de procesamiento de señal en formato de punto flotante, estas formas variables tienen poca importancia práctica.
- Para la implementación de **puntos fijos**, es importante entender sus **susceptibilidades relativas al ruido de cuantización**.
- Es importante que un diseñador digital entienda que la simple conversión de una implementación de punto flotante en punto fijo utilizando cualquier técnica de optimización de elección puede no producir resultados deseables y el sistema puede incluso volverse inestable.
- Por lo tanto, es esencial siempre seleccionar una forma que tenga **mínima sensibilidad a la cuantización de coeficientes** antes de que se realice la conversión de formato.
- Por lo tanto, el primer objetivo del diseño para una implementación de punto fijo de un algoritmo es elegir una forma que sea menos sensible al ruido de cuantización y minimizar los recursos de hardware para la implementación seleccionada.

Estructuras de Filtros Digitales

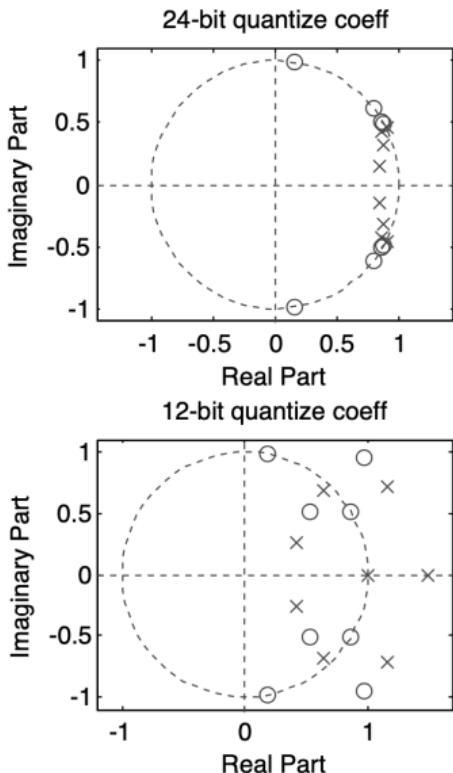
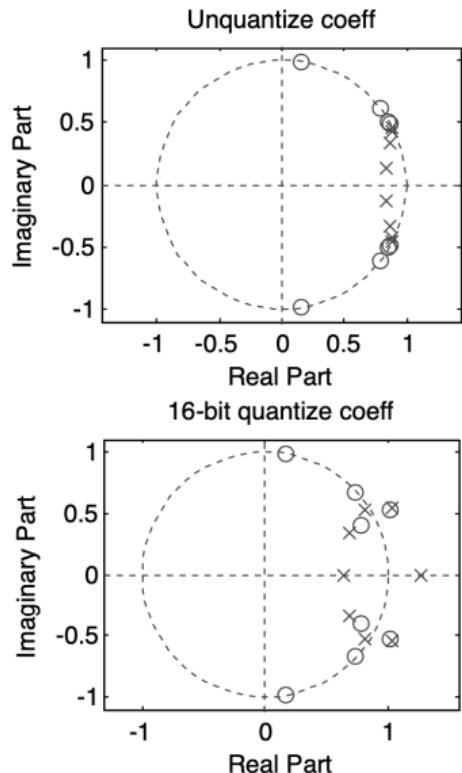
Estructuras en forma directa IIR

Cuantización de Coeficientes

- La cuantización de los coeficientes puede mover la posición de los polos y ceros y afectar la estabilidad y respuesta en frecuencia del filtro.
- Este efecto se puede observar en las siguientes figuras en donde se muestra la cuantización de los coeficientes del filtro considerando 24-bits, 16-bits y 12-bits.
- Como se observa en la gráfica de polos y ceros, el filtro es inestable para 16-bits y 12-bits.
- Este mismo filtro se puede implementar usando secciones de segundo orden (*Second-order-Sections - SoS*), donde podemos observar que son requeridos menos bits para mantener estable el mismo filtro.

Estructuras de Filtros Digitales

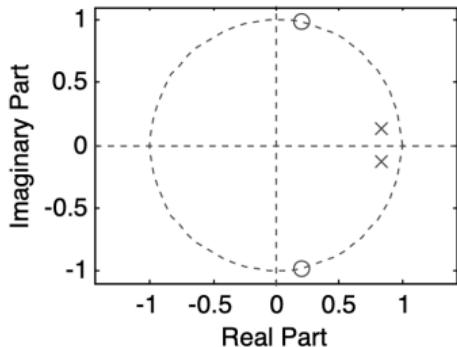
Estructuras en forma directa IIR



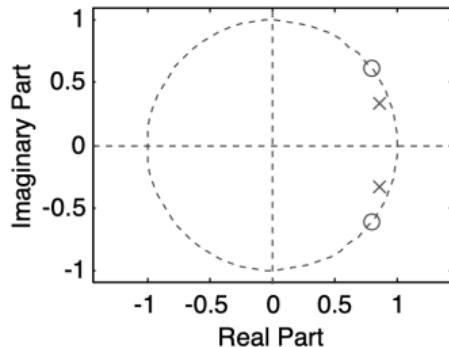
Estructuras de Filtros Digitales

Estructuras en forma directa IIR

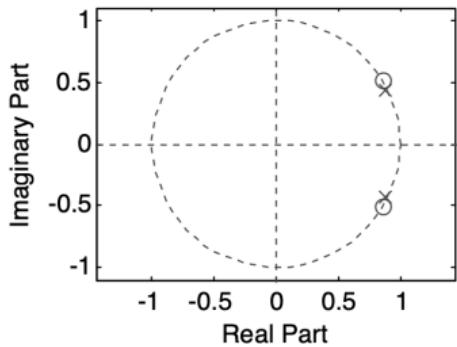
1st Section in Q2.10 Format



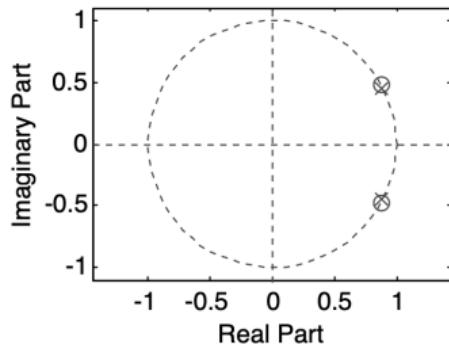
2nd Section in Q2.10 Format



3rd Section in Q2.10 Format

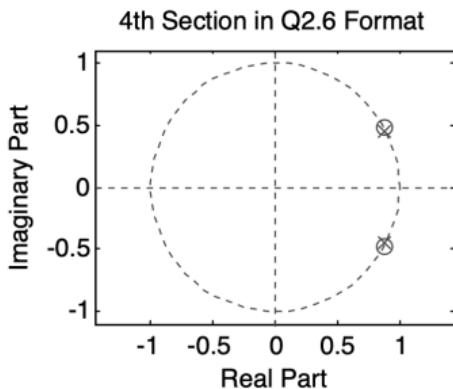
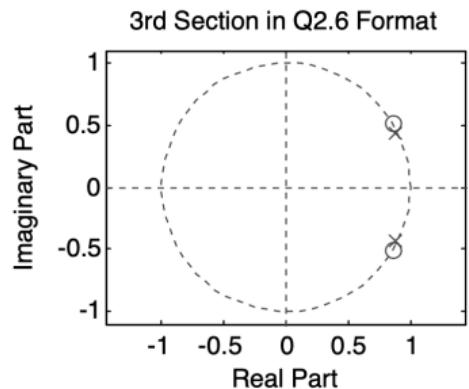
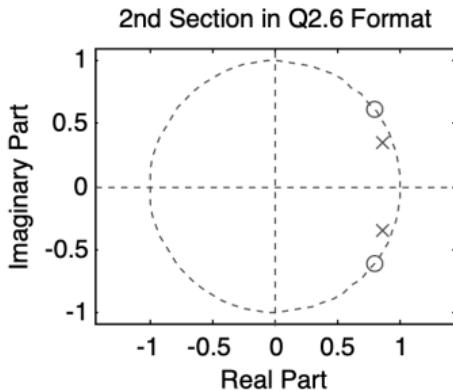
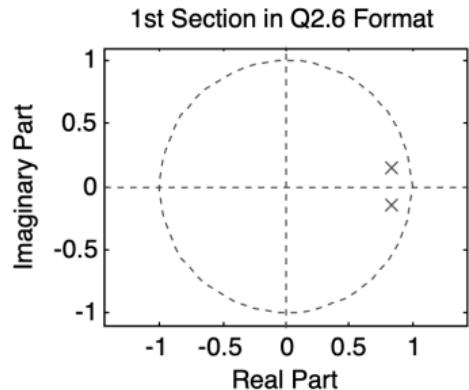


4th Section in Q2.10 Format



Estructuras de Filtros Digitales

Estructuras en forma directa IIR



Representaciones Numéricas



Representaciones Numéricas

Punto Flotante & Punto Fijo

Punto Flotante & Punto Fijo

- Desde una perspectiva de procesamiento de señales, se puede implementar un algoritmo usando formato de [punto fijo o flotante](#).
- El formato de [punto flotante](#) almacena un número en términos de mantisa y exponente.
- El hardware que soporta el formato de punto flotante, después de ejecutar cada cálculo, escala automáticamente la mantisa y actualiza el exponente para hacer que el resultado encaje en el número de bits requerido de una manera definida.
- Todas estas operaciones hacen que el hardware para operaciones en punto flotante sea más [costoso en términos de área y potencia](#).
- El hardware en [punto fijo](#), después de ejecutar un cálculo, no rastrea la posición del punto decimal y deja esta responsabilidad al desarrollador.
- El punto decimal se fija para cada variable y está predefinido.
- Al fijar el punto, una variable sólo puede tener un [rango fijo de valores](#).
- Como la variable está limitada, si el resultado de un cálculo cae fuera de este rango, se produce un desbordamiento (*overflow*).

Representaciones Numéricas

Punto Flotante & Punto Fijo

Punto Flotante & Punto Fijo

- Hay varias soluciones para manejar los desbordamientos (*overflows*) en aritmética de punto fijo.
- El manejo de desbordes requiere saturar el resultado a su máximo valor positivo o mínimo negativo que se puede asignar a una variable definida en formato de punto fijo.
- Esto se traduce en una reducción del desempeño o precisión.
- El diseñador puede fijar los lugares de puntos decimales para todas las variables, de modo que la disposición evite cualquier desbordamiento.
- Esto requiere que el diseñador realice pruebas con todos los datos posibles y observe los rangos de valores que todas las variables toman en la simulación.
- Conocer los rangos de todas las variables en el algoritmo hace que la determinación del punto decimal que evita el desbordamiento sea trivial.
- La implementación de algoritmos utilizando punto fijo reduce la complejidad en comparación con punto flotante.

Representaciones Numéricas

Tipos de Representaciones

- En aplicaciones de procesamiento de señales, las señales analógicas se digitalizan en un número discreto de N bits.
- Considerando N bits

$$\mathbf{a} = a_{N-1}a_{N-2}\dots a_2a_1a_0$$

- Los números pueden considerarse signados (*signed*) y no signados (*unsigned*).
- **No signados:** Los N bits se utilizan para representar la magnitud.
- **Signados:** El bit a_{N-1} se lo utiliza para representar el signo y el resto de los bits para la magnitud.
- Representación para los números signados
 - Complemento a 1
 - Magnitud y Signo
 - Dígito de signo canónico (*Canonic Sign Digit - CSD*)
 - Complemento a 2

Representaciones Numéricas

Complemento a 2

- En un número signado $\mathbf{a} = a_{N-1}a_{N-2}\dots a_2a_1a_0$, el bit mas significativo (*Most Significant Bit - MSB*) a_{N-1} representa el signo.
- Los números positivos se representan con el cero y los negativos con el uno.

$$\left. \begin{array}{l} a = \sum_{i=0}^{N-2} a_i 2^i, \text{ para } a \geq 0 \\ a = -2^{N-1} + \sum_{i=0}^{N-2} a_i 2^i, \text{ para } a < 0 \end{array} \right\} a = -2^{N-1} a_{N-1} + \sum_{i=0}^{N-2} a_i 2^i$$

- La representación sin signo equivalente de números negativos viene dada por

$$2^N - |\mathbf{a}|$$

- Donde $|\mathbf{a}|$ es igual al valor absoluto del número negativo \mathbf{a} .

Ejemplo

- -9 en 5 bits en complemento a 2 es 10111.
- Número equivalente de -9 es $2^5 - |-9| = 23$.

Representaciones Numéricas

Definiciones

- **Rango:** Diferencia entre el mayor número positivo y el menor número negativo representables.
- **Precisión:** Número total de valores distintos representables, que se determina por la cantidad de bits usados en la representación.
- **Resolución:** Distancia mínima entre dos valores consecutivos representables.
- **Exactitud:** Cercanía entre el valor real y su representación, determinada por la magnitud de la diferencia entre ambos.
- **Rango Dinámico:** Cociente entre el máximo valor absoluto representable y el mínimo valor absoluto (distinto de cero) representable.

Representaciones Numéricas

Escalado

- Al implementar algoritmos usando aritmética de precisión finita, a veces es importante evitar el desbordamiento (*overflow*) ya que añade un error que es igual al rango dinámico completo del número.

■ Reducción

- Para evitar el desbordamiento, los números se reducen (*scaled-down*).
- Cuando un número tiene bits de signo redundantes, puede reducirse descartándolos.
- Esta reducción de bits no afectará el valor del número.

■ Extensión de signo

- En diseños digitales a veces también se requiere extender el signo de un número de N bits a un número de M bits para $M > N$.
- El signo se extiende $M - N$ bits.
- El valor del número no cambia pero si su representación equivalente.

Ejemplos

- Reducción:** El número -2 como número signado en complemento a dos de 8 bits es 8'b11111110. En este caso contiene seis bits de signo redundantes que pueden descartarse, pudiendo representarlo sólo con 2 bits como 2'b10.
- Extensión de signo:** El número -2 como número binario de 4 bits es 4'b1110. Extendiéndolo a un número de 8 bits es 8'b11111110.

Formato de Punto Flotante



Formato de Punto Flotante

Concepto

- La representación de punto flotante es conveniente para números con gran rango dinámico.
- Basado en el número de bits, hay dos representaciones en el estándar IEEE 754: 32 bits en simple precisión y 64 bits en doble precisión.
- Esta norma se utiliza casi exclusivamente en plataformas informáticas y diseños de hardware que soportan la aritmética de punto flotante.
- Un número de punto flotante normalizado x se almacena en tres partes
 - El signo s
 - Exponente e
 - Mantisa m

$$x = (-1)^s \times 1 \times m \times 2^{e-b}$$

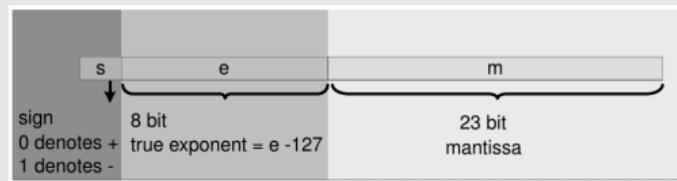
- s representa el signo del número y m da la magnitud normalizada con un 1 en la posición MSB, siendo ese bit implícito y no almacenado con el número. Para valores normalizados, m representará entonces una magnitud mayor a 1 y menor que 2.
- El formato IEEE almacena el exponente e como un número sesgado, es decir un número positivo del que se resta un sesgo constante b para obtener el exponente positivo o negativo real.

Formato de Punto Flotante

Representaciones estándar

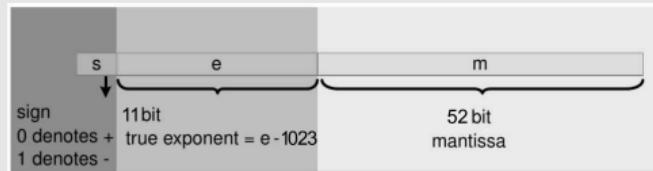
■ Punto flotante de simple precisión (32 bits)

- 1 bit de signo s
- 8 bits de exponente e (0 a 255). El sesgo b vale 127.
- 23 bits de mantisa m



■ Punto flotante de doble precisión (64 bits)

- 1 bit de signo s
- 11 bits de exponente e (0 a 2047). El sesgo b vale 1023.
- 52 bits de mantisa m



Formato de Punto Flotante

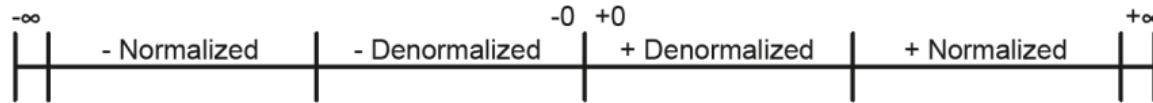
Ejemplo: representación en punto flotante de simple precisión

- Número: -12.25
- Signo y Magnitud (binario): -00001100.01
- Moviendo el punto: $-1100.01 \times 2^0 = -1.10001 \times 2^3$ (normalizado)
- Bit de signo (s): 1
- Mantisa (m): 10001000_00000000_0000000
- Exponente (e): $3 + 127 = 130 = 1000_0010$
- 32-bits:1_10000010_10001000_00000000_0000000

Formato de Punto Flotante

Valores especiales del exponente e

- Para el caso de valores pequeños donde el exponente e vale 0, se utiliza únicamente el valor de la mantisa para representar el número. A este rango de números se los trata como *desnormalizados* y el bit implícito de la mantisa valdrá 0. Fuera de este rango siempre se considerará la operación de normalización en la representación.
- En los casos en que el número de bits del exponente no sean suficientes para expresar valores normalizados positivos o negativos, se utilizan dos combinaciones especiales que se representan como $+\infty$ y $-\infty$.
- Para el caso de operaciones cuyo resultado no sea posible de calcular (ej. $0/0$), se utiliza una combinación especial representada como NaN (Not a Number).



value	s	e	m
$+\infty$	0	11111111	00000000000000000000000000000000
$-\infty$	1	11111111	00000000000000000000000000000000
NAN	1	11111111	10000000000000000000000000000000

Formato de Punto Flotante

Adición Aritmética

Pasos

- 1) Añadir el bit implícito de la mantisa. Este bit es 1 o 0 para los números normalizados y desnormalizados, respectivamente.
- 2) Desplazar las mantisas de 1) con exponente mas pequeño e_s hacia la derecha por $e_l - e_s$, donde e_l es el mayor de los dos exponentes. Este cambio puede lograrse proporcionando unos pocos bits de guarda a la derecha para una mejor precisión.
- Si alguno de los operandos es negativo, tomar el complemento a 2 de la mantisa de 2) y luego sumar las dos mantisas. Si el resultado es negativo, tomar de nuevo el complemento a dos del resultado.
- 3) Normalizar la suma de nuevo al formato IEEE ajustando la mantisa y cambiando apropiadamente el valor del exponente e_l . Comprobar si hay “underflow” y “overflow” del resultado.
- 4) Redondear o truncar la mantisa resultante al número de bits definido en el estándar.

Formato de Punto Flotante

Adición Aritmética

Ejemplo

- Suma en 10 bits de precisión ($s : 1, e : 4, m : 5, b : 7$)
- $9.250 \rightarrow 1001.01x2^0 \rightarrow 1.00101x2^3 \rightarrow 0_1010_00101$
- $4.625 \rightarrow 100.101x2^0 \rightarrow 1.00101x2^2 \rightarrow 0_1001_00101$
- Alineo las comas y agrego un bit:
 $1.00101x2^3 \rightarrow 1.001010x2^3$
 $1.00101x2^2 \rightarrow 0.100101x2^3$
- Sumo las mantisas :
 $1.001010 + 0.100101 = 1.101111$
- Trunco a 5 bits el valor de la mantisa:
 $13.750 \rightarrow 1.10111x2^3 \rightarrow 0_1010_10111$

Formato de Punto Flotante

Multiplicación Aritmética

Pasos

- 1) Añadir los dos exponentes e_1 y e_2 . Como el sesgo inherente en los dos exponentes se suma dos veces, restar el sesgo una vez de la suma para obtener el exponente resultante e en el formato correcto.
- 2) Colocar el 1 implícito de la mantisa si los operandos se guardan como números normalizados. Multiplicar las mantisas como números sin signo para obtener el producto, y aplicar una XOR a los dos bits de signo para obtener el signo del producto.
- 3) Normalizar el producto si es necesario para adecuarlo al formato IEEE. Comprobar si el resultado hace “underflow” o “overflow”.
- 4) Redondear o truncar la mantisa a la cantidad de bits definida por el estándar.

Formato de Punto Flotante

Multiplicación Aritmética

Ejemplo

- Multiplicación en 10 bits de precisión ($s : 1, e : 4, m : 5, b : 7$)
- $0.25 \rightarrow 0.01x2^0 \rightarrow 1.0x2^{-2} \rightarrow 0_0101_00000$
- $0.25 \rightarrow 0.01x2^0 \rightarrow 1.0x2^{-2} \rightarrow 0_0101_00000$
- Sumo los exponentes considerando la transformación y resto un sesgo:
 $(-2+7) + (-2+7) - 7 = 3 \rightarrow 3 - 7 = -4$
- Determino el signo:
 $0 \text{ XOR } 0 = 0$
- Multiplico las mantisa como valores *unsigned*:
 $1.00000x1.00000 = 1.0000000000$
- Truncamos
 $0.0625 \rightarrow 1.00000x2^{-4} \rightarrow 0_0011_00000$

Formato de Punto Flotante

Aplicación en HW

- Operar en punto flotante requiere de tres operaciones: ajuste de exponente, cálculo matemático y normalización.
- En este sentido, la ejecución de operaciones de punto flotante en HW es muy costosa en términos de potencia, área y performance.
- Sólo tiene sentido en aplicaciones que involucren algoritmos donde mantener la exactitud intacta sea realmente necesario.
- En generaciones avanzadas de FPGAs o ASICs de muy alta densidad se incluye hardware para mapear algoritmos en punto flotante si así se requiere.
- En sistemas digitales de comunicaciones y procesamiento de señales, normalmente el diseñador comienza desarrollándolo usando aritmética de punto flotante de doble precisión en lenguajes como C++, Python o Matlab. Luego de verificarlo funcionalmente y obtener la performance deseada, las variables y constantes de simulación se convierten a punto fijo para evaluar el desempeño frente a la cuantización y determinar los recursos necesarios para su implementación en hardware.

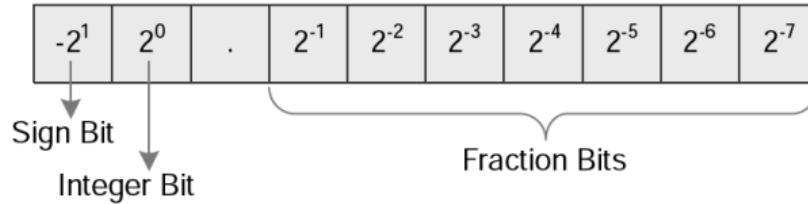
Aritmética de Punto Fijo

Aritmética de Punto Fijo

Concepto

- La representación en punto fijo de un número $S(NB, NBF)$ consiste en la implementación de NB bits, de los cuales NBI representan la parte entera, NBF representan la parte fraccional y el bit MSB puede considerarse como el signo en casos que se opere con aritmética signada.
- Un número en complemento a dos de un número en punto fijo es equivalente a:

$$-b_{n-1}2^{n-1} + b_{n-2}2^{n-2} + \cdots + b_12^1 + b_22^2 + \cdots + b_m2^m$$



Aritmética de Punto Fijo

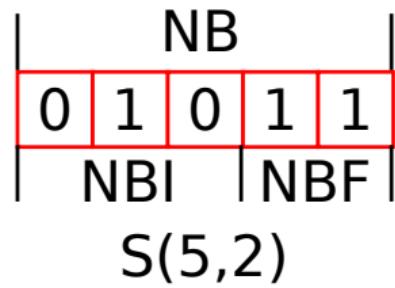
Conversión de Punto Flotante a Punto Fijo

- La conversión a punto fijo requiere de un cuidado análisis. Esto se debe a que impacta directamente sobre el desempeño y complejidad del diseño.
- La complejidad se reduce utilizando un número pequeño de longitud de palabra en las variables del código, pero esto afecta negativamente al rendimiento numérico del algoritmo y añade **ruido de cuantización**.
- Un número de punto flotante se convierte simplemente en formato de punto fijo de $S(NB, NBF)$ tomando los NB bits de la parte entera (saturando si fuese necesario) y luego los NBF bits de la parte fraccionaria truncando el resto de los bits con o sin redondeo.
- Desde el punto de vista de la implementación, un número en punto fijo se puede ver como un numero binario entero cuya posición decimal está implícita. Esto hace que el diseñador deba recordar el formato de cada número para poder realizar operaciones sobre el mismo de forma adecuada.

Aritmética de Punto Fijo

Representación Binaria

- Representación de un número **S(NB,NBF)**
- Signado (S) o no signado (U)
- Número de bits totales (NB)
- Número de bits enteros ($NBI=NB-NBF$)
- Número de bits fraccionales (NBF)



Aritmética de Punto Fijo

Representación no signada U(NB,NBF)

- El valor de una número binario x de NB-bits se representa por la expresión:

$$x = \left(\frac{1}{2^{NBF}} \right) \sum_{n=0}^{NB-1} 2^n b_n \quad (37)$$

Rango

- El rango de valores que se pueden representar es
 $0 \leq U(NB, NBF) \leq (2^{NB} - 1)/2^{NBF} = 2^{NBI} - 2^{-NBF}$
- Ejemplo de representación binaria $b_5b_4b_3b_2b_1b_0.b_{-1}b_{-2}$ donde b_k es el peso 2^k .

Aritmética de Punto Fijo

Notación - unsigned

U(8,6)

X	X	X	X	X	X	X	X

U(8,10)

0	0	X	X	X	X	X	X	X	X

U(8,0)

X	X	X	X	X	X	X	X

U(8,-2)

X	X	X	X	X	X	X	X	0	0

Aritmética de Punto Fijo

Determinar el valor decimal y el rango

- $U(8,2) = 0x8A$ (1000_1010b)

- **Valor:** $(1/2^2)(2^7 + 2^3 + 2^1) = 34.5$

- **Rango:** $0 \leq U(8,2) \leq 2^6 - 2^{-2} = 63.75$

- $U(8,0) = 0x8A$

- **Valor:** $(1/2^0)(2^7 + 2^3 + 2^1) = 138$

- **Rango:** $0 \leq U(8,0) \leq 2^8 - 2^0 = 255$

- $U(16,18) = 0x04BC$ (0000_0100_1011_1100b)

- **Valor:** $(1/2^{18})(2^{10} + 2^7 + 2^5 + 2^4 + 2^3 + 2^2) = 0.0046233413085938$

- **Rango:** $0 \leq U(16,18) \leq 2^{-2} - 2^{-18} = 0.2499961853027$

Aritmética de Punto Fijo

Representación signada S(NB,NBF)

- El valor de una número binario x de NB-bits se representa por la expresión:

$$x = \left(\frac{1}{2^{NBF}} \right) \left[-2^{NB-1} b_{NB-1} + \sum_{n=0}^{NB-2} 2^n b_n \right] \quad (38)$$

Rango

- El rango de valores que se pueden representar es

$$-2^{NB-1-NBF} \leq S(NB, NBF) \leq 2^{NB-1-NBF} - 2^{-NBF}$$

$$-2^{NBI-1} \leq S(NB, NBF) \leq 2^{NBI-1} - 2^{-NBF}$$

Aritmética de Punto Fijo

Notación - signed

S(8,6)

S	X	X	X	X	X	X	X
---	---	---	---	---	---	---	---



S(8,10)

S	S	S	X	X	X	X	X	X	X
---	---	---	---	---	---	---	---	---	---



S(8,0)

S	X	X	X	X	X	X	X
---	---	---	---	---	---	---	---



S(8,-2)

S	X	X	X	X	X	X	X	0	0
---	---	---	---	---	---	---	---	---	---



Aritmética de Punto Fijo

Determinar el valor decimal y el rango

- S(8,2) = 0x8A (1000_1010b)

- Valor:** $(1/2^2)(-2^7 + 2^3 + 2^1) = -29.5$

- Rango:** $-32 \leq S(8,2) \leq 2^5 - 2^{-2} = 31.75$

- S(8,0) = 0x8A

- Valor:** $(1/2^0)(-2^7 + 2^3 + 2^1) = -118$

- Rango:** $-128 \leq S(8,0) \leq 2^7 - 2^0 = 127$

- S(16,18) = 0x04BC (0000_0100_1011_1100b)

- Valor:** $(1/2^{18})(2^{10} + 2^7 + 2^5 + 2^4 + 2^3 + 2^2) = 0.0046233413085938$

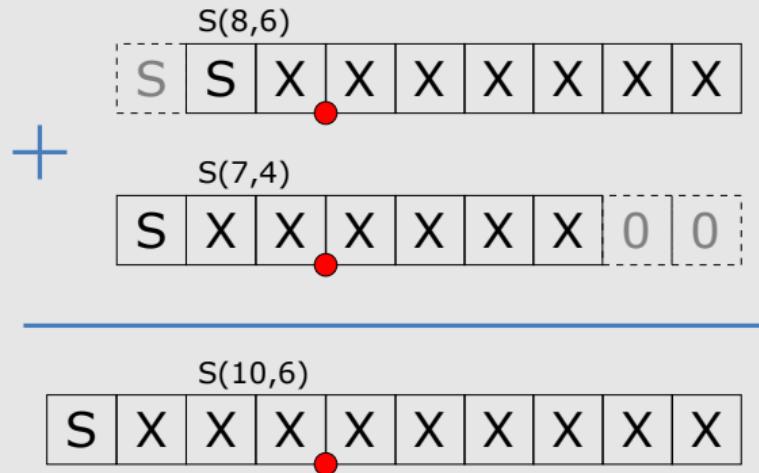
- Rango:** $-0.125 \leq S(16,18) \leq 2^{-3} - 2^{-18} = 0.124996185302734375$

Aritmética de Punto Fijo

Operaciones

Suma/Resta

- En las operaciones de suma y resta, se agrega siempre un bit adicional en la parte entera del resultado (acarreo).
- Es importante alinear las comas, por lo que se considera siempre el mayor tamaño de la parte entera y fraccionaria entre los sumandos.



Aritmética de Punto Fijo

Operaciones

Multiplicación

- El valor de salida tendrá un tamaño igual a la suma de los bits de ambos operadores.

$$\begin{array}{r} S(8,6) \\ \boxed{\begin{array}{ccccccccc} S & X & X & X & X & X & X & X & X \end{array}} \\ \times \\ \boxed{\begin{array}{ccccccccc} S & X & X & X & X & X & X & X \end{array}} \end{array}$$

$$S(15,10)$$
$$\boxed{\begin{array}{cccccccccccccccc} S & X & X & X & X & X & X & X & X & X & X & X & X & X & X & X \end{array}}$$

Aritmética de Punto Fijo

Operaciones: crecimiento de bits

Truncado y Redondeo

- Las operaciones de truncado y redondeo se aplican sobre la parte fraccional del número en punto fijo al limitar la cantidad de bits.

		NBI				NBF				
Inicial	$S(8,4)$	0	1	1	1	0	1	1	0	7.375
Truncado	$S(6,2)$	0	1	1	1	0	1	1	0	7.25
	$S(8,4)$	0	1	1	1	0	1	1	0	7.375
								1		
Redondeo	$S(8,4)$	0	1	1	1	1	0	0	0	
Truncado	$S(6,2)$	0	1	1	1	1	0	0	0	7.5

- 1/0: Signo.

Aritmética de Punto Fijo

Operaciones: crecimiento de bits

Overflow y Saturación

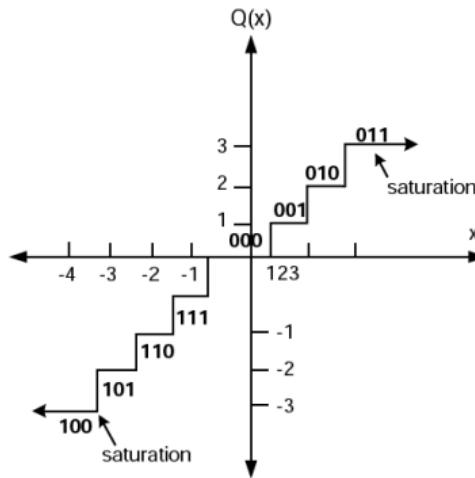
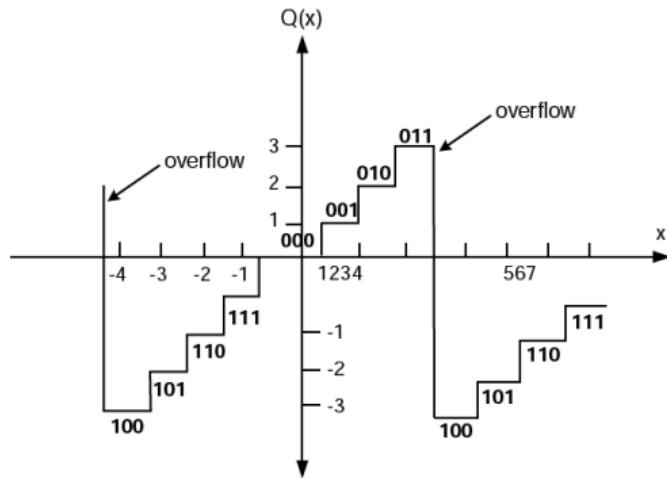
- Las operaciones de *overflow* y saturación se aplican sobre los bits más significativos del número en punto fijo al limitar la cantidad de bits.

		NBI				NBF				
Inicial	$S(8,4)$	0	1	1	1	0	1	1	0	7.375
Overflow	$S(6,4)$			1	1	0	1	1	0	-0.625
Saturación	$S(6,4)$			0	1	1	1	1	1	1.9375

- 1/0: Signo.

Aritmética de Punto Fijo

Overflow y Saturación



Aritmética de Punto Fijo

Suma

		NBI				NBF				
Acarreo		1	1	1	1					
	$S(6,2)$	1	1	1	1	1	0	0	0	-0.5
	$S(8,4)$	0	0	1	1	1	0	1	1	7.375
Máx. Res.	$S(9,4)$	0	0	1	1	0	1	1	1	6.875
Truncado	$S(7,2)$	0	0	1	1	0	1	1		6.75
Redondeo	$S(7,2)$	0	0	1	1	1	0	0		7.00
Overflow	$S(7,4)$			1	1	0	1	1	1	-1.125
Saturación	$S(7,4)$			0	1	1	1	1	1	3.9375

- 1/0: Signo.
- 1/0: Expansión de signo.
- 1: Acarreo.
- 0: Completa con ceros.

Aritmética de Punto Fijo

Multiplicación - $U(.) \times U(.)$

	$U(4,2)$				1	1	0	1	3.25
	$U(4,2)$				1	0	1	1	2.75
		0	0	0	0	1	1	0	1
		0	0	0	1	1	0	1	-
		0	0	0	0	0	0	-	-
		0	1	1	0	1	-	-	-
Máx. Res.	$U(8,4)$	1	0	0	0	1	1	1	8.9375

- (-): Corrimiento por multiplicación.
- 0: Completa con ceros.

Aritmética de Punto Fijo

Multiplicación - $S(\cdot) \times U(\cdot)$

	$S(4,2)$				1	1	0	1	-0.75
	$U(4,2)$				1	0	1	1	2.75
		1	1	1	1	1	0	1	
		1	1	1	1	1	0	-	
		0	0	0	0	0	0	-	
		1	1	1	0	1	-	-	
Máx. Res.	$S(8,4)$	1	1	0	1	1	1	1	-2.0625

- (-): Corrimiento por multiplicación.
- 1/0: Signo.
- 1/0: Expansión de signo.

Aritmética de Punto Fijo

Multiplicación - $U(\cdot) \times S(\cdot)$

	$U(4,2)$					1	1	0	1	3.25
	$S(4,2)$					1	0	1	1	-1.25
		0	0	0	0	1	1	0	1	
		0	0	0	1	1	0	1	-	
		0	0	0	0	0	0	-	-	
Complemento 2's		1	0	0	1	1	-	-	-	
Máx. Res.	$S(8,4)$	1	0	1	1	1	1	1	1	-4.0625

- (-): Corrimiento por multiplicación.
- 1/0: Signo.
- 0: Completa con ceros.

Aritmética de Punto Fijo

Multiplicación - $S(\cdot) \times S(\cdot)$

	$S(4,2)$					1	1	0	1	-0.75
	$S(4,2)$					1	0	1	1	-1.25
		1	1	1	1	1	1	0	1	
		1	1	1	1	1	1	0	1	-
		0	0	0	0	0	0	-	-	
Complemento 2's		0	0	0	1	1	-	-	-	
Máx. Res.	$S(8,4)$	0	0	0	0	1	1	1	1	0.9375

- (-): Corrimiento por multiplicación.
- 1/0: Signo.
- 1/0: Expansión de signo.