Unidad 1.C. Representación finita de números reales en punto flotante

Dr. Ing. Hernán Garrido

Control y sistemas
Universidad Nacional de Cuyo, Facultad de Ingeniería

carloshernangarrido@gmail.com

Noviembre de 2023

Contenidos

Motivación

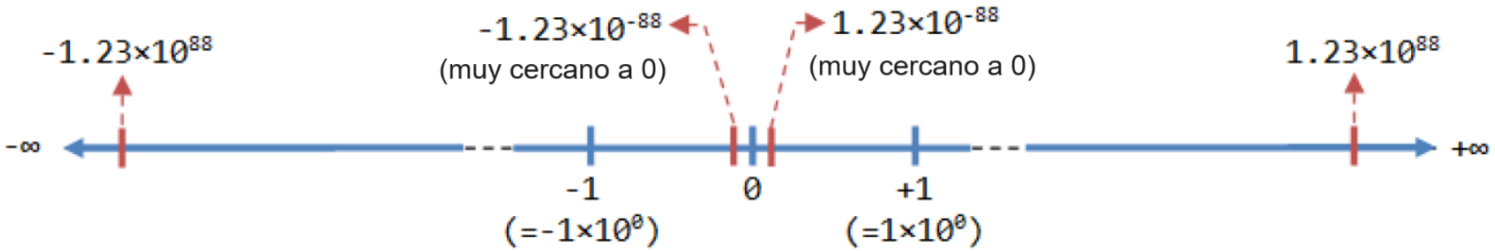


Figura: Números en punto flotante (decimal).

Un número en punto flotante típicamente se expresa en notación científica en la forma:

$$(-1)^S \cdot s \cdot B^e$$

donde

- S es el bit de signo,
- s es una fracción llamada mantisa o significando,
- e es un exponente sesgado, y
- B es 10 para base decimal o 2 para binarios.

Contenidos

Estándar IEEE 754-2008

- Han habido varios formatos en el pasado, por ejemplo IBM, DEC, MIL-STD 1750A, los cuales asignaban distinta cantidad de bits a F y E.
- La mayoría de las computadoras modernas adoptan el formato IEEE 754
  - Primera versión: 1985.
  - Última versión: 2019.

	Binary formats ($B = 2$)			
Parameter	Binary 16	Binary 32	Binary 64	Binary 128
p, digits	$10 + 1$	$23 + 1$	$52 + 1$	$112 + 1$
e_{max}	+15	+127	+1023	+16383
e_{min}	-14	-126	-1022	-16382
Common name	Half precision	Single precision	Double precision	Quadruple precision

Figura: Formatos binarios de la IEEE 754-2008

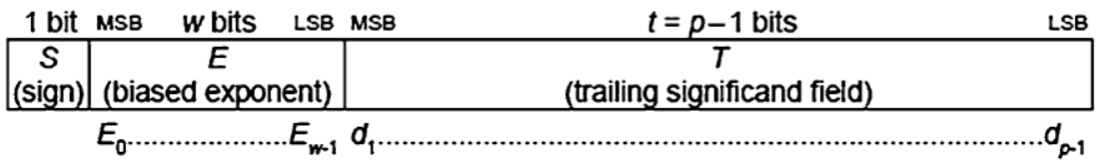
Lo que se guarda y su significado

Las cadenas de k bits están compuestas por tres campos:

- 1 bit de signo S,
- Un exponente sesgado de w bits $E = e + \text{bias}$ (E es un entero sin signo),
- Los $p - 1$ bits finales del significando; el bit faltante se codifica en el exponente (primer bit oculto).

Table 12.2 Binary interchange format parameters

Parameter	Binary16	Binary32	Binary64	Binary128	Binary{k} ($k \geq 128$)
k, storage width in bits	16	32	64	128	Multiple of 32
p, precision in bits	11	24	53	113	$k - w$
e_{max}	15	127	1,023	16,383	$2^{(k-p-1)} - 1$
$bias$, $E - e$	15	127	1,023	16,383	e_{max}
w, exponent field width	5	8	11	15	Round$(4 \cdot \log_2 k) - 13$
t, trailing significand bits	10	23	52	112	$k - w - 1$



Contenidos

Representación normalizada y denormalizada

- La representación de números en punto flotante podría no ser única;
- por ejemplo, $1101.01_2 \cdot (2^0) = 110.101_2 \cdot (2^1) = 11.0101_2 \cdot (2^2)$

Representación normalizada: Se codifica con $E > 0$

El bit oculto de la mantisa es implícitamente igual a 1 y se ajusta la parte fraccionaria.

$$s = 1.T = 1 + T_{t-1}2^{-1} + T_{t-2}2^{-2} + ... + T_1 2^{-t+1} + T_0 2^{-t}$$

$$(-1)^S \cdot s \cdot B^e = (-1)^S \cdot s \cdot 2^{E-\text{bias}}$$

Representación denormalizada: Se codifica con $E = 0$

El bit oculto de la mantisa es implícitamente igual a 0.

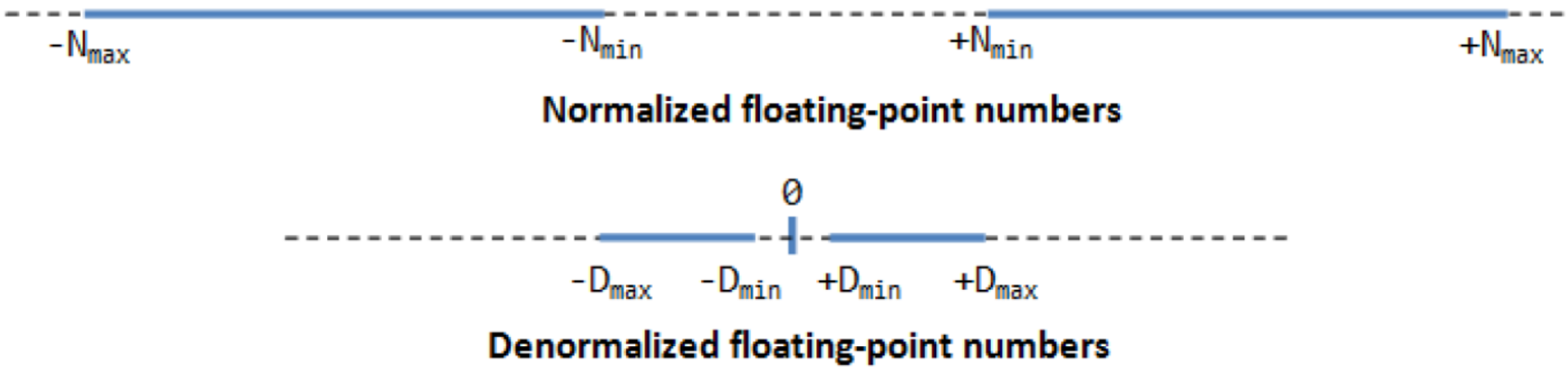
$$s = 0.T = 0 + T_{t-1}2^{-1} + T_{t-2}2^{-2} + ... + T_1 2^{-t+1} + T_0 2^{-t}$$

$$(-1)^S \cdot s \cdot B^e = (-1)^S \cdot s \cdot 2^{-\text{bias}}$$

Representación normalizada: Características

Características

- Representación:
  - incompleta, como toda representación finita; pero
  - única, gracias a forzar el 1 antes del punto decimal
- Multiplicar y dividir por 2 es trivial
  - Simplemente se suma o se resta 1 al exponente (ya que la base es 2)
- ¡Auto-rango!
  - ... como toda notación exponencial. La notación científica y la representación en punto flotante son casos particulares de notación exponencial.



Representación normalizada: Ejemplos

Ejemplo 1: $3215.020002 \cdot 2 = 6430.040004$

Decimal Value Entered: 6430.040004

Single precision (32 bits):

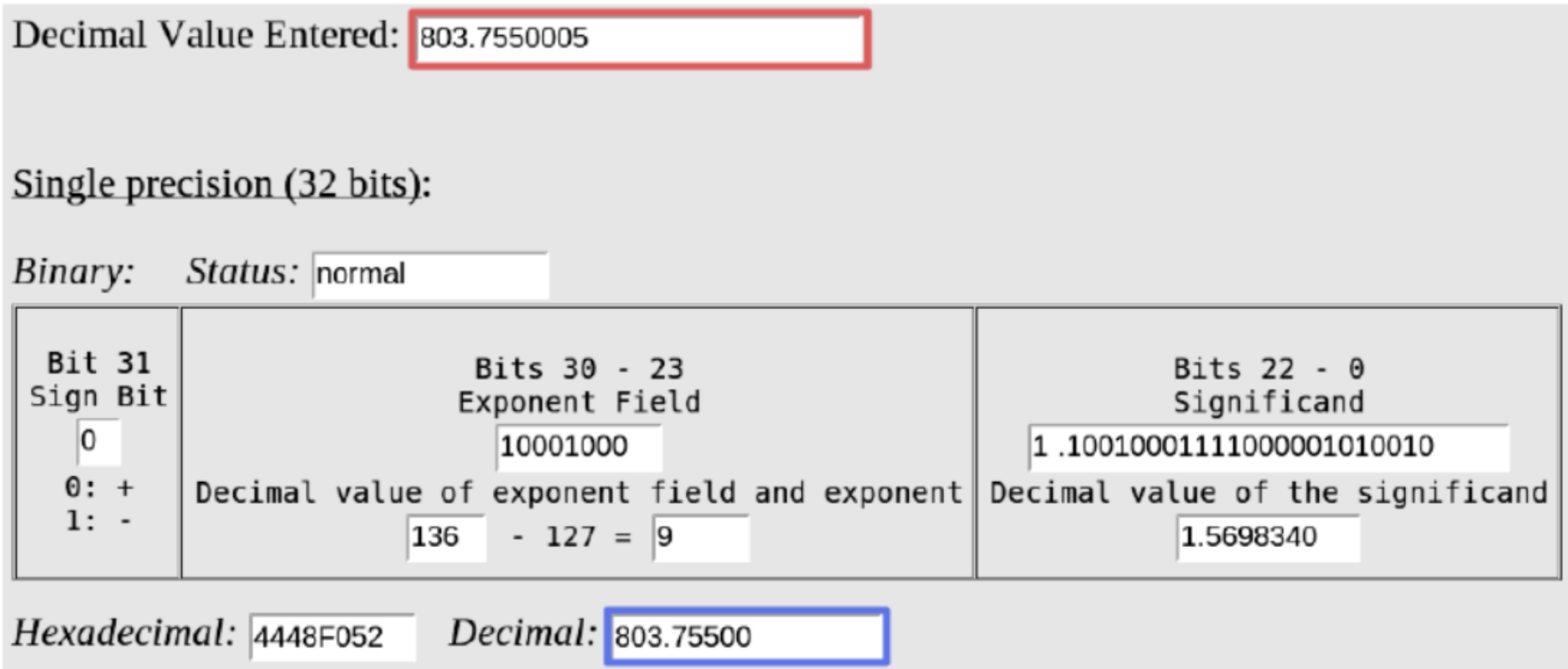
Binary: *Status:* normal

Bit 31 Sign Bit	Bits 30 - 23 Exponent Field	Bits 22 - 0 Significand
0	10001011	1 .10010001111000001010010
0: + 1: -	Decimal value of exponent field and exponent	Decimal value of the significand
	139 - 127 = 12	1.5698340

Hexadecimal: 45C8F052 *Decimal:* 6430.0400

Representación normalizada: Ejemplos

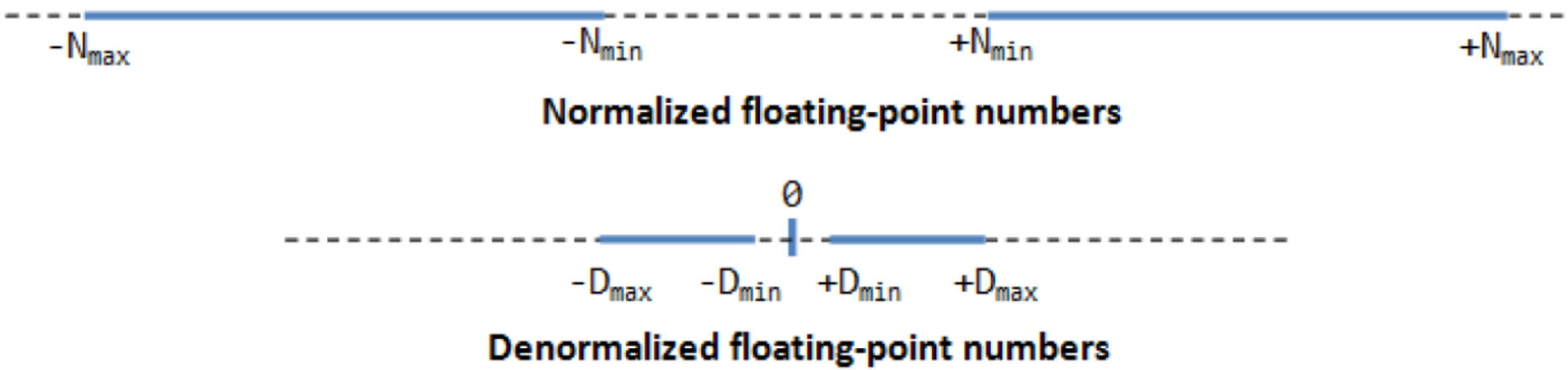
Ejemplo 2: 3215.020002 / 4 = 803.7550005



Representación denormalizada: Características

Características

- Representación:
  - incompleta, como toda representación finita; pero
  - única, gracias a forzar que el exponente sea constante.
- Se puede representar el 0
  - Con un 1 implícito a la izquierda del punto no se podía.
- Sirve para representar número pequeños:
  - El exponente es el más chico posible, y
  - pueden haber varios 0s luego del punto y antes del primer 1.



Representación denormalizada: Ejemplos

Ejemplo 3: $-3.4 \cdot 10^{-39}$

Decimal Value Entered: -3.4e-39

Single precision (32 bits):

Binary: *Status:* denormalized

Bit 31 Sign Bit	Bits 30 - 23 Exponent Field	Bits 22 - 0 Significand
1	00000000	0.10010100000101110100001
0: + 1: -	Decimal value of exponent field and exponent	Decimal value of the significand
	0 - 127 = -127	0.5784800

Hexadecimal: 802505D1 *Decimal:* -3.3999999e-39

Contenidos

Números especiales

- Zero = cero: $E = 0$, $T = 0$. Tiene dos representaciones
  - -0 ($S = 1$)
  - +0 ($S = 0$)
- Infinity = Inf = infinito: $E = 111...1$, $T = 0$. Tiene dos representaciones
  - -Inf ($S = 1$)
  - +Inf ($S = 0$)
- Not a number = NaN = no-número: $E = 111...1$, $T \neq 0$. Resultados de operaciones que no son números reales, por ejemplo $0/0$.

```
1  % En MATLAB
2  a = 1/0;                % a = Inf
3  b = exp(1000);          % b = Inf
4  c = log(0);             % c = -Inf
5  d = -1/0;               % d = -Inf
6  e = 0/0;                % e = NaN
7  f = Inf/Inf;            % f = NaN
```

Contenidos

Esquemas de redondeo: Definiciones

Unidad de menor precisión

Unidad de menor precisión = unit of least precision = unit of last place = ulp $\approx$ eps (Matlab) = exactitud relativa de punto flotante
Sean f', f'' dos valores en punto flotante consecutivos, y $x \in \mathbb{R}$:
Si $f' \leq x \leq f''$, entonces:

$$\text{ulp}(x) = f'' - f'$$

Si $f' \leq |x| \leq f''$, entonces:

$$\text{eps}(x) = f'' - |x|$$

Esquema de redondeo

Es el criterio y/o procedimiento para redondear el resultado x de una operación en punto flotante a ya sea a f' o a f''.

Esquemas de redondeo

Sea $f' \leq x \leq f''$:

- Truncado = rendodeo hacia 0 = cropping

$$\text{round}(x) = \begin{cases} f' & si \quad x > 0 \\ f'' & si \quad x < 0 \\ 0 & si \quad x = 0 \end{cases}$$

- Rendondeo hacia más infinito

$$\text{round}(x) = f''$$

- Rendondeo hacia menos infinito

$$\text{round}(x) = f'$$

- Rendondeo al más próximo

$$\text{round}(x) = \begin{cases} f' & si \quad x < f' + \frac{\text{ulp}(x)}{2} \\ \\ f'' & si \quad x \geq f' + \frac{\text{ulp}(x)}{2} \end{cases}$$

Contenidos

Rango Dinámico

Definición

$$\mathrm{DR_{dB}} = 20 \log_{10} \left(\frac{\text{mayor valor posible}}{\text{menor valor posible}} \right) \text{ [dB]}$$

Para números de punto flotante:

$$\mathrm{DR_{dB}} \approx 6.02 \cdot 2^{w}$$

donde w es el número de bits del campo del exponente E.
Para números de punto flotante con precisión simple (32 bits):

$$\mathrm{DR_{dB}} \approx 6.02 \cdot 2^{8} \approx 1541\,\mathrm{dB}$$

Para números de punto fijo con precisión simple (32 bits):

$$\mathrm{DR_{dB}} \approx 6.02 \cdot 31 \approx 186\,\mathrm{dB}$$

Contenidos

Precisión en punto fijo

```
% En MATLAB
% Fixed-point quantizer
q = quantizer('fixed','floor','saturate',[5 1]);
% [wordlength fractionlength]
u = linspace(-15,15,1000);
y1 = quantize(q,u);
plot(u,y1); title(tostring(q))
```

Precisión en punto fijo: Resolución constante

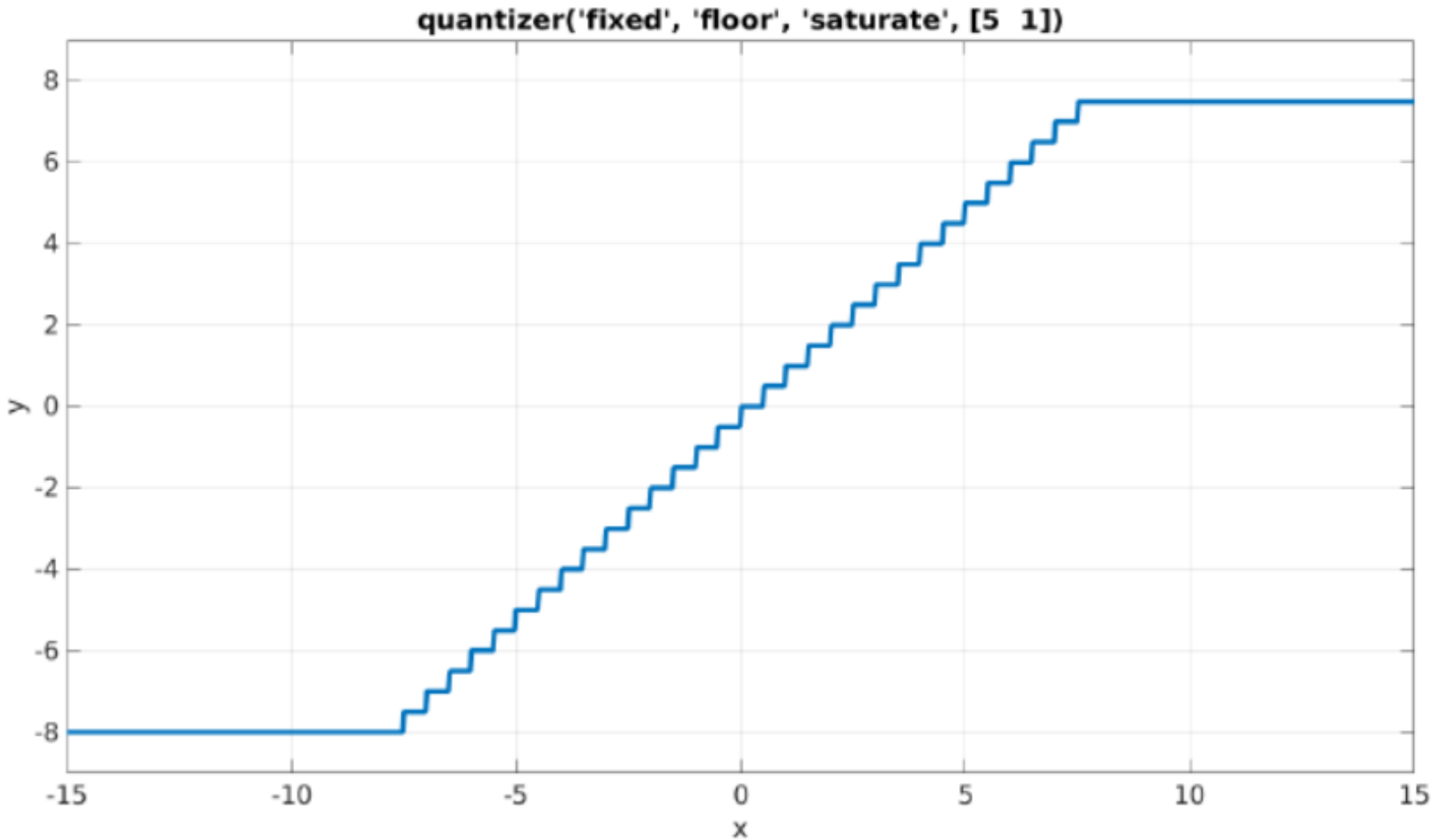


Figura: 5 bits: 1 para la parte fraccionaria, 1 para el signo, y 3 para la parte entera.

Precisión en punto flotante

```
% En MATLAB
% Floating-point quantizer
q = quantizer([5 3],'float', 'nearest');
% [wordlength exponentlength]
y2 = quantize(q,u);
plot(u,y2); title(tostring(q))
```

Precisión en punto flotante: Resolución variable

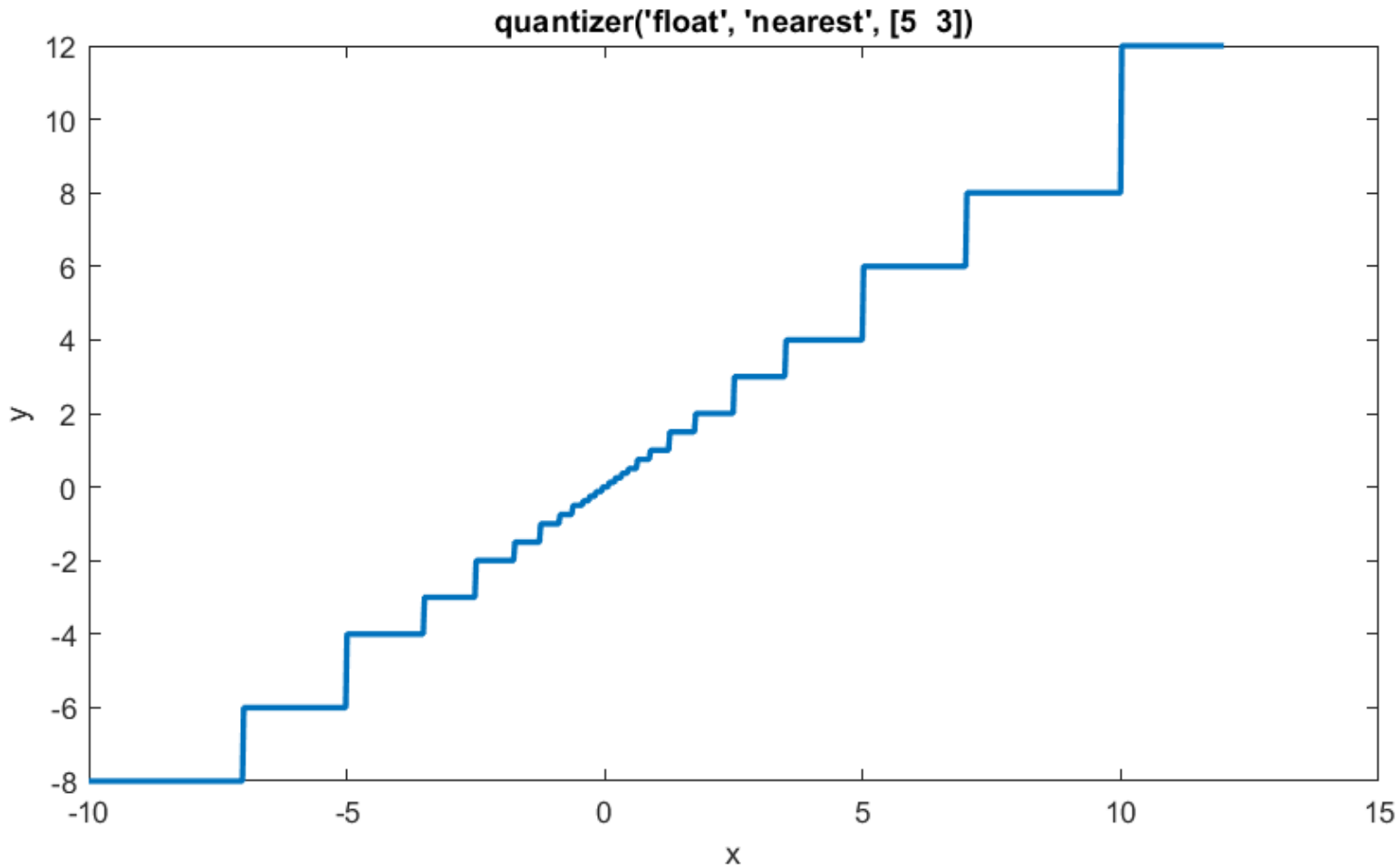


Figura: 5 bits: 3 para el exponente, 1 para el signo, y 1 para la parte fraccionaria de la mantisa ($1.1_2 \cdot 2^{2^3-5} = 1.5 \cdot 2^3 = 12$).

Contenidos

Problemas de precisión

Cuando la misma operación involucra números muy grandes y muy pequeños, se pierde el valor del pequeño.

```
% En MATLAB
a = (2^53 + 1) - 2^53;
% a = 0;
if (a == 0)
    disp( Turn off nuclear reactor )
else
    disp( Do not turn off nuclear reactor )
end

x = 0;
t = tan(x) - sin(x)/cos(x)
t = 0
x = 1;
t = tan(x) - sin(x)/cos(x)
t = 2.2204e-16 % eps(1)
```

Suma de dos números de orden similar

Ejemplo: Sumar $0.5 + (-0.4375)$ utilizando 4 bits para la mantisa

$$0.5_{10} = 0.1000_2 \cdot 2^0 = 1.0000_2 \cdot 2^{-1}$$

$$-0.4375_{10} = -0.0111_2 \cdot 2^0 = -1.1100_2 \cdot 2^{-2}$$

Hacer coincidir los exponentes al mayor: Aplicar n corrimientos a -0.4375 donde $n = (\text{exponente1} - \text{exponente2}) = -1 - (-2) = 1$.

$$-0.4375 = -1.1100_2 \cdot 2^{-2} = -0.1110_2 \cdot 2^{-1}$$

Sumar las mantisas:

$$(1.0000_2 - 0.1110_2) \cdot 2^{-1} = 0.0010_2 \cdot 2^{-1}$$

Normalizar la suma, verificando el overflow/underflow:

$$0.0010_2 \cdot 2^{-1} = 1.0000_2 \cdot 2^{-4} = 0.0625$$

Si $e_{min} \leq -4 \leq e_{max}$, no hay overflow ni underflow.
Redondear la suma: Como cabe en 4 bits, no es necesario redondear.

Suma de dos números de orden diferente

Ejemplo: Sumar $10^{10} + 1500$ con IEEE-754 de 32 bits ($p = 23 + 1, w = 8$)

$$10000000000_{10} = 1.00101010000001011111001_2 \cdot 2^{33}$$

$$1500_{10} = 1.01110111000000000000000_2 \cdot 2^{10}$$

Hacer coincidir los exponentes al mayor: Aplicar n corrimientos al 1500 donde $n = 33 - 10 = 23$.

$$1500_{10} \approx 0.00000000000000000000001_2 \cdot 2^{33} = 1 \cdot 2^{10} = 1024$$

Sumar las mantisas:

$$(1.00101010000001011111001_2 + 0.00000000000000000000001_2) \cdot 2^{33}$$

Normalizar la suma, verificando el overflow/underflow:

$$1.00101010000001011111010_2 \cdot 2^{33} = 1.16415333710 \cdot 2^{33} = 10000001024$$

Si $-127 \leq 33 \leq 127$, no hay overflow ni underflow.
Redondear la suma: Como cabe en 23 bits, no es necesario redondear.

Comparación entre punto flotante y punto fijo

Ventajas

Aspecto	**Punto Flotante**	**Punto Fijo**
Precisión	Mayor rango dinámico	Resolución constante
Costos	Menor tiempo de desarrollo	Menor costo de producción

Bibliografía

1. IEEE-SA Standards Board. IEEE Standard for Floating-Point Arithmetic. ISBN 978-0-7381-5752-8. Approved 12 June 2008. New York, NY, USA.
2. Jean-Pierre Deschamps, Gustavo D. Sutter, and Enrique Cantó. Floating Point Arithmetic. Guide to FPGA Implementation of Arithmetic Functions, Chapter 12. Springer, 2012.