Organización del computador

Sistemas de representación

Sistemas de numeración

Números reales

- Los números reales **no** pueden ser representados totalmente, no solo por un problema de magnitud (existen infinitos números reales) sino también de precisión (existen reales que tienen infinitos decimales)
- Los registros de un procesador tienen un tamaño finito en bits y por ende solo pueden representar una cantidad de información finita
- En general se representan como: **a) punto fijo**, pudiendo ser magnitud signada o complemento a 2, o **b) punto flotante**.

Sistemas de numeración

Números reales: representación de punto fijo

- Se representan mediante la expresión:

$$(a_n \ldots a_2 a_1 a_0 . a_{-1} a_{-2} \ldots a_{-m})_2 = a_n * 2^n + \ldots + a_2 * 2^2 + a_1 * 2 + a_0 + a_{-1} * 2^{-1} + a_{-2} * 2^{-2} + \ldots + a_{-m} * 2^{-m}$$

  b es un entero mayor que 1 y para todo $-m \leq i \leq n$, vale que $0 \leq a_i < 2$

- La distancia entre dos número reales es 2^{-m} dejando de ser un conjunto continuo y pasando a ser discreto
- Se representa la parte entera con signo usando alguna de las representaciones vistas (magnitud signada, complemento a 1 ó complemento a 2) y la parte fraccionaria mediante un cambio de base

Sistemas de numeración

Truncamiento y redondeo

- **Problema:** Debemos representar un número con n dígitos decimales en un sistema con m dígitos decimales siendo m < n
- **Solución 1 (Truncamiento):** Descartamos los dígitos fraccionarios de orden mayor a m. El error en peor caso es de 1 bit
- **Solución 2 (Redondeo):** Descartamos los dígitos fraccionarios de orden mayor a m y se suma 1 al bit menos significativo en caso de que el siguiente sea 1. El error en peor caso es de 0.5 bit

Sistemas de numeración

Notación científica

- El número n se escribe como $m*10^e$, donde m y e son números enteros con signo

$-725.832 = -7.25832 \times 10^2 = -725.832 \times 10^0$

$3.14 = 0.314 \times 10^1 = 3.14 \times 10^0$

$0.000001 = 0.1 \times 10^{-5} = 1.0 \times 10^{-6}$

$1941 = 0.1941 \times 10^4 = 1.941 \times 10^3$

- Para unificar la notación se recurre a la normalización exigiendo que $0.1 \leq f < 1$

Sistemas de numeración

Números reales: representación de punto flotante

- El número $(n)_b$ se escribe como m*b^e, donde m y e son números enteros con signo; luego se los puede representar con un par (m, e) tal que m se encuentra normalizado
- Mantisa y exponente se representan usando alguno de los métodos vistos anteriormente pues son magnitudes con signo

Sistemas de numeración

Números reales: representación de punto flotante normalizado

- El número $(n)_{10}$ veamos cómo obtener su representación en punto flotante normalizado en base 2:

  1) Determinar el exponente: encontrar e tal que

  $$2^{e-1} \leq |n| < 2^e$$

  2) Hallar la mantisa: por definición es $|m|=|n|/2^e$

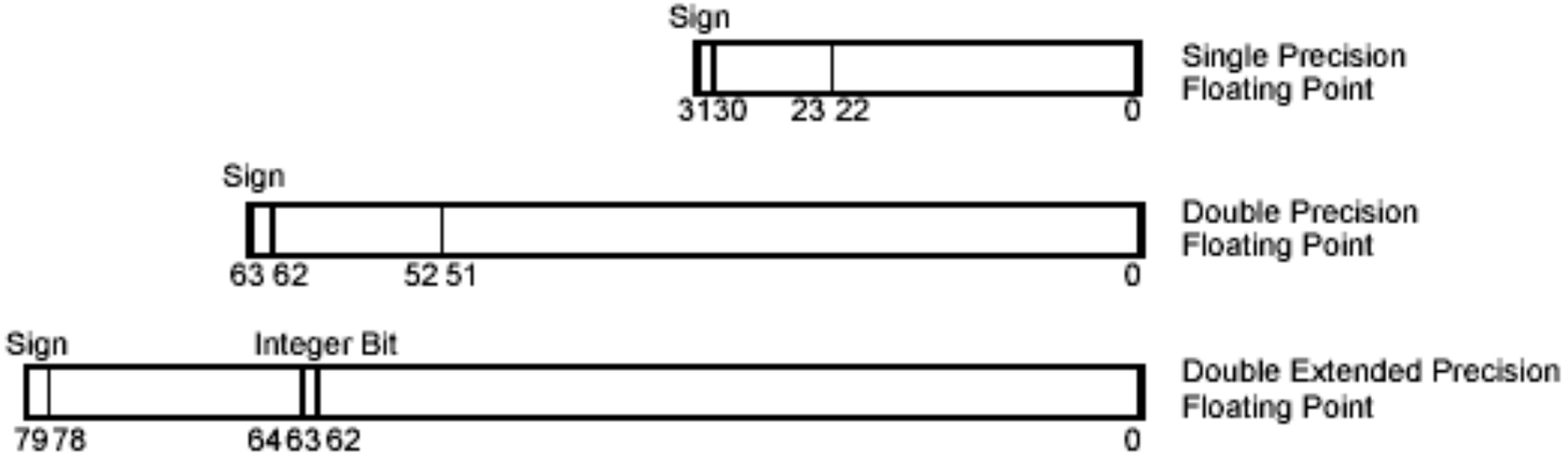
  3) Representar m y e en base 2

  4) Construir la representación: usualmente es signo, k bits para el exponente (en notación exceso) y k' bits para la mantisa

Sistemas de numeración

IEEE 754

- Institute of Electrical and Electronics Engineers
- El standard IEEE 754 regula la representación de números en punto flotante binario



Data Type	Length	Precision (Bits)	Approximate Normalized Range	
			Binary	Decimal
Single Precision	32	24	2^{-126} to 2^{127}	1.18×10^{-38} to 3.40×10^{38}
Double Precision	64	53	2^{-1022} to 2^{1023}	2.23×10^{-308} to 1.79×10^{308}
Double Extended Precision	80	64	2^{-16382} to 2^{16383}	3.37×10^{-4932} to 1.18×10^{4932}

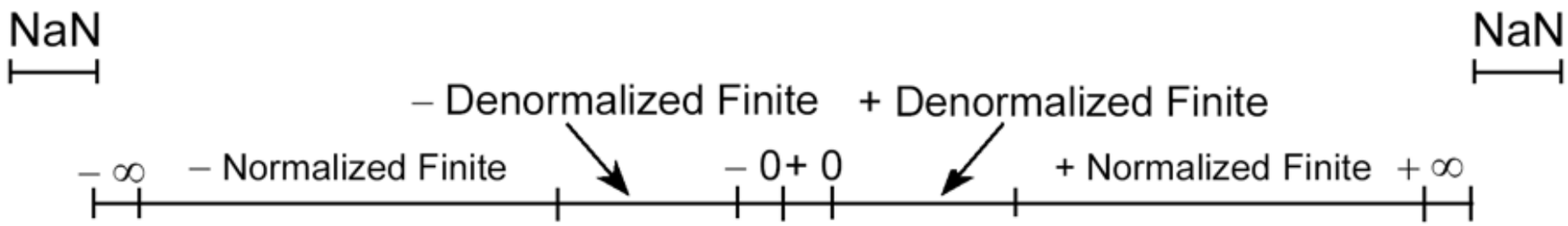
Sistemas de numeración

Problemas de representación del punto flotante normalizado

- Es standard IEEE 754 especifica números quedo satisfacen los criterios de normalización explicados anteriormente:
  - -> números denormalizados
  - -> ceros con signo
  - -> infinitos con signo
  - -> NaN (Not a number)
  - -> números indefinidos

Sistemas de numeración

Problemas de representación del punto flotante normalizado



Real Number and NaN Encodings For 32-Bit Floating-Point Format

S	E	Sig[1]		S	E	Sig[1]	
1	0	0.000...	– 0	+ 0	0	0	0.000...
1	0	0.XXX...[2]	– Denormalized Finite	+Denormalized Finite	0	0	0.XXX...[2]
1	1...254	1.XXX...	– Normalized Finite	+Normalized Finite	0	1...254	1.XXX...
1	255	1.000...	– ∞	+ ∞	0	255	1.000...
X[3]	255	1.0XX...[2]	SNaN	SNaN	X[3]	255	1.0XX...[2]
X[3]	255	1.1XX...	QNaN	QNaN	X[3]	255	1.1XX...

NOTES:

1. Integer bit of fraction implied for single-precision floating-point format.
2. Fraction must be non-zero.
3. Sign bit ignored.

Sistemas de numeración

Problemas de representación del punto flotante normalizado

- **Ceros signados**: se interpreta como 0 pero el signo proporciona información sobre el resultado que hubiera dado la operación si la precisión hubiera sido suficiente
- **Normalizados**: son expresables con una mantisa normalizada y un exponente entre -126 y 127
- **Denormalizados**: resultan de números que se aproximan demasiado a 0 y -126 no es suficiente para normalizar la mantisa

Operation	Sign	Exponent*	Significand
True Result	0	–129	1.01011100000...00
Denormalize	0	–128	0.10101110000...00
Denormalize	0	–127	0.01010111000...00
Denormalize	0	–126	0.00101011100...00
Denormal Result	0	–126	0.00101011100...00

* Expressed as an unbiased, decimal number.

- **Infinitos signados**: números cuya magnitud es más grandes que cualquier número representable, también poseen signo para identificara el rango del resultado
- **NaNs**: resultan de una operación inválida y hay de dos tipos, Signaling (lanza una excepción) y Quiet (son propagadas)

Caracteres

ASCII

- El código ASCII utiliza 7 bits para representar los caracteres, aunque inicialmente empleaba un bit adicional (bit de paridad) para detectar errores en la transmisión
- A menudo se llama incorrectamente ASCII a otros códigos, como el estándar ISO-8859-1 que que utiliza 8 bits para proporcionar caracteres adicionales usados en idiomas distintos al inglés
- Define códigos para 33 caracteres no imprimibles, de los cuales la mayoría son caracteres de control obsoletos que tienen efecto sobre como se procesa el texto, más otros 95 caracteres imprimibles
- Todos los sistemas informáticos actuales utilizan el código ASCII o una extensión compatible para representar textos

Caracteres

ISO 8859-1

ISO-8859-1																
	x0	**x1**	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**	**x9**	**xA**	**xB**	**xC**	**xD**	**xE**	**xF**
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	l	}	~	DEL
8x	PAD	HOP	BPH	NBH	IND	NEL	SSA	ESA	HTS	HTJ	VTS	PLD	PLU	RI	SS2	SS3
9x	DCS	PU1	PU2	STS	CCH	MW	SPA	EPA	SOS	SGCI	SCI	CSI	ST	OSC	PM	APC
Ax	NBSP	¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	SHY	®	¯
Bx	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Caracteres

Unicode

- Standard promovido por la industria para universalizar la codificación de de texto en cualquier idioma
  especialmente por a necesidad de representación mutilenguaje y las limitaciones de los standards existentes
- Soportado a alto nivel por sistemas operativos, formatos de distribución de documentos como XML y lenguajes de programación como Java, C#, Python, etc.

Caracteres

UTF-8

- UTF-8 (8-bit Unicode Transformation Format): es una norma de transmisión de longitud variable para caracteres en Unicode
- Utiliza grupos de bytes para representar el standard Unicode de distintos alfabetos
- Usa de 1 a 4 bytes por caracter pero su transferencia se hace sobre 8 bits lo que lo hace compatible con los servicios de correo

Caracteres

UTF-8 - Ventajas

- La secuencia de bytes de un caracter nunca es prefijo de otra más larga de otro caracter lo que las hace fáciles de manipular
- El primer byte codifica a longitud de la secuencia correspondiente al caracter
- Está diseñado para no generar conflicto con sistemas *legacy* y así generar incompatibilidades (no utiliza bytes de control ASCII o ISO-8859-1)

Caracteres

UTF-8 - Desventajas

- Es de longitud variable haciendo más complicada su implementación,
- Necesita una capa abstracción más entre el software de usuario / el sistema operativo y la arquitectura de la computadora
- La necesidad de analizadores de UTF-8 podrían introducir errores que antes no existían