Math 230
Assembly Language Programming
(Computer Organization)

Numeric Data

Lecture 2

Decimal Numbers

- Recall base 10
  - 3582

    $= 3000 + 500 + 80 + 2$

    $= 3\times10^3 + 5\times10^2 + 8\times10^1 + 2\times10^0$

Positional Notation

- Number in base (or "radix") B has B distinct digits
- The position of the digit is used to determine the value
- Position notation starts with zero
- 32 digit number:

$$d_{31} d_{30} \ldots d_2 d_1 d_0$$

$$d_{31} \times B^{31} + d_{30} \times B^{30} + \ldots + d_1 \times B^1 + d_0 \times B^0$$

Positional Notation (cont'd)

- Binary Numbers
- $11101_{two} = 11101_2 =$ 0b11101
- NB: 5 digit binary converts to 2 digit decimal

Review: Positional Notation

What digits can be used in base 5?

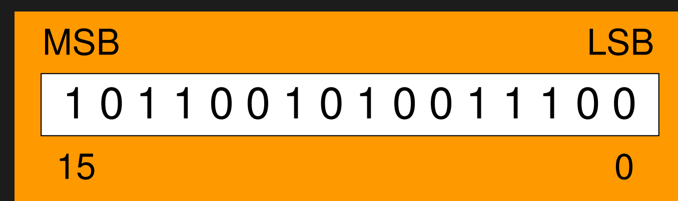
What value is represented by 321_{five}?

What digits can be used in base 8?

base16?

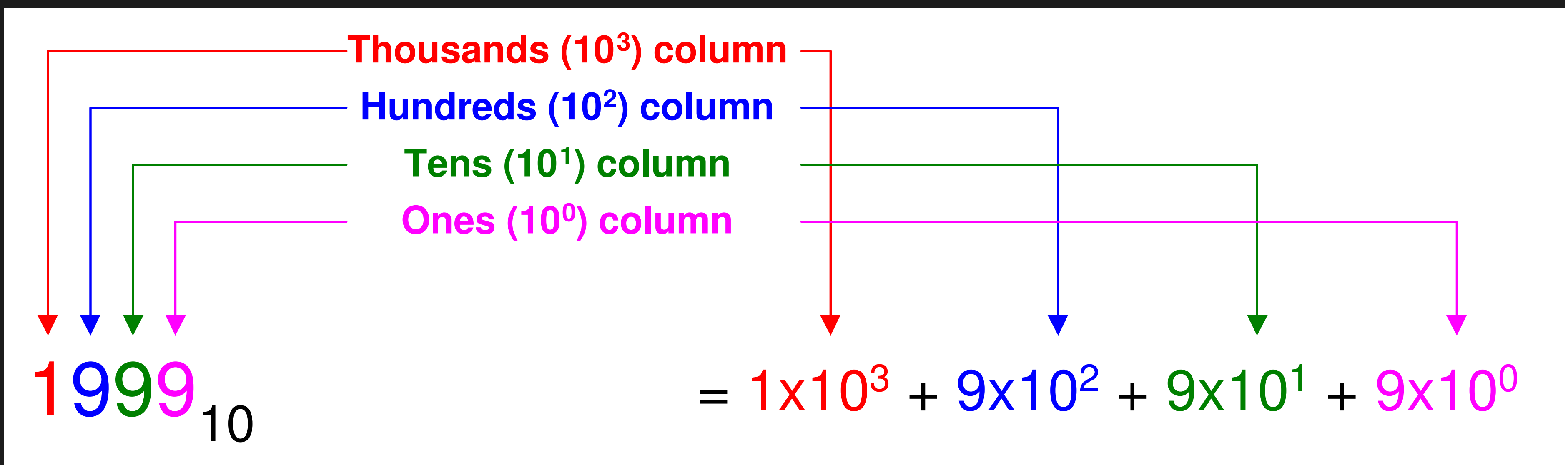
Binary Numbers

- Digits are 1 and 0
  - 1 = true
  - 0 = false
- MSB – most significant bit
- LSB – least significant bit
- Bit numbering:



Base-10 (decimal) arithmetic

- Uses the ***ten*** numbers from 0 to 9
- Each column represents a power of ***10***

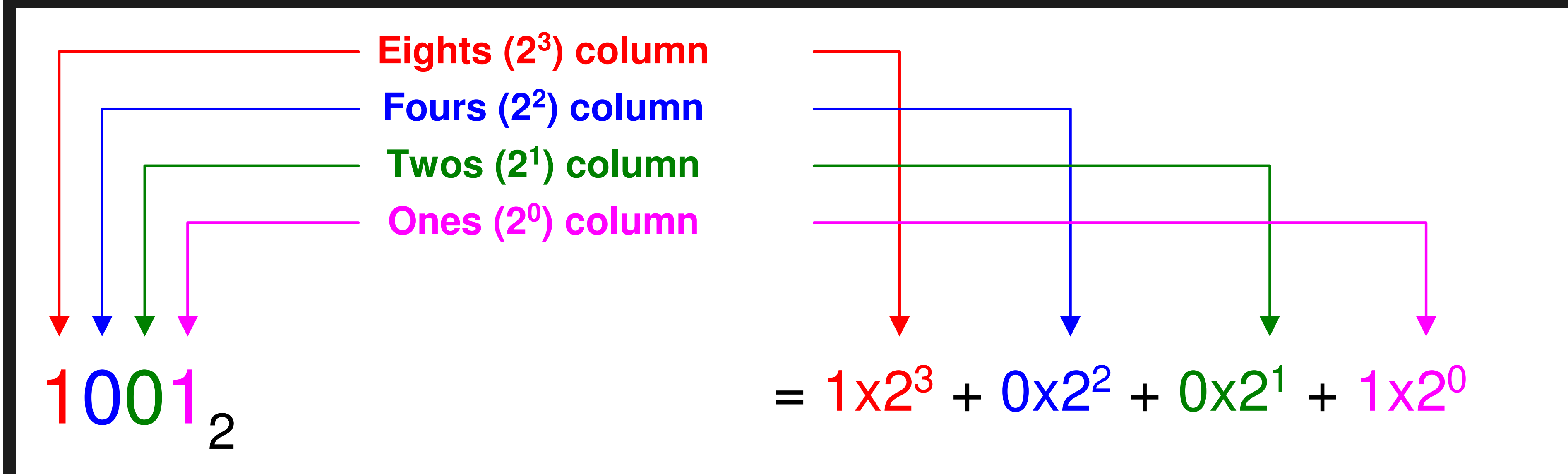


Base-2 (binary) arithmetic

- Uses the ***two*** numbers from 0 to 1
- Every column represents a power of ***2***

Base-2 (binary) arithmetic

- Uses the ***two*** numbers from 0 to 1
- Every column represents a power of ***2***



Translating Binary to Decimal

Weighted positional notation shows how to calculate the decimal value of each binary bit:

$$dec = (D_{n-1} \times 2^{n-1}) + (D_{n-2} \times 2^{n-2}) + ... + (D_1 \times 2^1) + (D_0 \times 2^0)$$

D = binary digit

binary 00001001 = decimal 9:

$$(1 \times 2^3) + (1 \times 2^0) = 9$$

Converting from base-2 to base-10

32	*16*	*8*	*4*	*2*	*1*	
		1	0	0	1	=
		1	0	1	1	=
	1	0	1	0	1	=
1	1	1	1	1	1	=

Converting from base-10 to base-2 (on the fly)

64	*32*	*16*	*8*	*4*	*2*	*1*		
							=	16
							=	55
							=	75
							=	84

Converting from base-10 to base-2 (using division)

- Repeatedly divide the decimal integer by 2. Each remainder is a binary digit in the translated value:

Division	Quotient	Remainder
37 / 2	18	1
18 / 2	9	0
9 / 2	4	1
4 / 2	2	0
2 / 2	1	0
1 / 2	0	1

37 = 100101

Addition

Base-10

	1	9	9	8
+			1	1
	2	0	0	9

Base-2

	1	0	1	1
+			1	1
	1	1	1	0

Binary Addition

- Starting with the LSB, add each pair of digits, include the carry if present.

Practice binary arithmetic

1	0	1	1
+		1	1

1	1	1
+		1

Hexadecimal (Base16)

- Hexadecimal
  - Hex = 6, decimal = 10 => 16 digits
  - A, B, C ,D, E, F
- In C: 0xA34

- Conversion: Binary ← → Hex
  - 4 binary digits required to represent largest possible hex digit
  - 1 hex digit replaces 4 binary digits
  - 4 binary digits is called a nibble
  - 2 nibbles is a byte
    - e.g. 1001 1100 0011 = 0x9C3

Decimal vs Hexadecimal vs Binary

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

  101101

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

  101101

  = 0010 1101

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

  101101

  = 0010 1101

  = 0x2D

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

  101101

  = 0010 1101

  = 0x2D

  0x3F9

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Decimal vs Hexadecimal vs Binary

- Examples

  1101 1110 0001

  = 0xDE1

  101101

  = 0010 1101

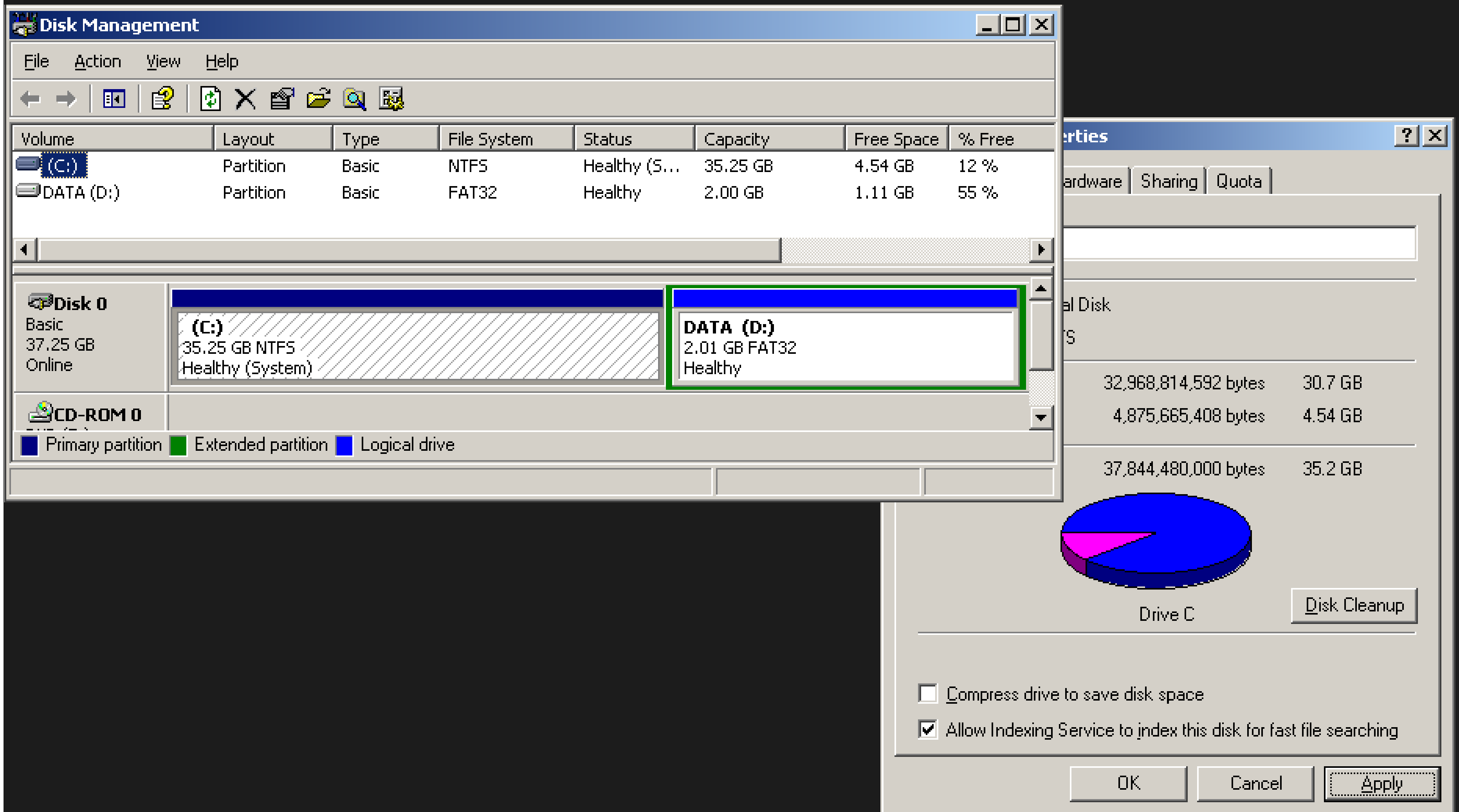
  = 0x2D

  0x3F9

  = 0011 1111 1001

00	0	0000
01	1	0001
02	2	0010
03	3	0011
04	4	0100
05	5	0101
06	6	0110
07	7	0111
08	8	1000
09	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

How much is a GB or a MB?



Kilo, Mega, Giga, Tera, Peta, Exa, Zetta, Yotta

physics.nist.gov/cuu/Units/binary.html

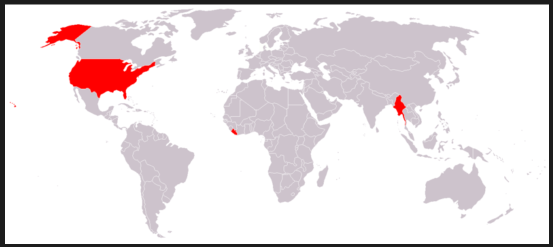
- Common use prefixes (all SI, except K [= k in SI])

Name	Abbr	Factor	SI size
Kilo	K	2^{10} = 1,024	10^{3} = 1,000
Mega	M	2^{20} = 1,048,576	10^{6} = 1,000,000
Giga	G	2^{30} = 1,073,741,824	10^{9} = 1,000,000,000
Tera	T	2^{40} = 1,099,511,627,776	10^{12} = 1,000,000,000,000
Peta	P	2^{50} = 1,125,899,906,842,624	10^{15} = 1,000,000,000,000,000
Exa	E	2^{60} = 1,152,921,504,606,846,976	10^{18} = 1,000,000,000,000,000,000
Zetta	Z	2^{70} = 1,180,591,620,717,411,303,424	10^{21} = 1,000,000,000,000,000,000,000
Yotta	Y	2^{80} = 1,208,925,819,614,629,174,706,176	10^{24} = 1,000,000,000,000,000,000,000,000

- Confusing! Common usage of “kilobyte” means 1024 bytes, but the “correct” SI value is 1000 bytes
- Hard Disk manufacturers & Telecommunications are the only computing groups that use SI factors, so what is advertised as a 30 GB drive will actually only hold about 28 x 2^{30} bytes, and a 1 Mbit/s connection transfers 10^{6} bps.

The metric system (SI prefix)

- Units of 1000 or 10^3 have proved useful in the metric system



- SI prefixes:
  - kilo (10^3), mega(10^6), giga(10^9), tera(10^{12})

1 Km is actually 1000 m, which makes sense

Binary Prefix

- Units of 1024 or 2^{10} have proved useful in the world of computers

- New prefixes have been created:
  - kibi (2^{10}), mebi (2^{20}), gibi (2^{30}), tebi(2^{40})

what HDD mfrs have been calling MB should be MiB

Prefixes for Binary Multiples

Factor	Name	Symbol	Origin	Derivation
2^{10}	kibi	Ki	kilobinary: $(2^{10})^1$	kilo: $(10^3)^1$
2^{20}	mebi	Mi	megabinary: $(2^{10})^2$	mega: $(10^3)^2$
2^{30}	gibi	Gi	gigabinary: $(2^{10})^3$	giga: $(10^3)^3$
2^{40}	tebi	Ti	terabinary: $(2^{10})^4$	tera: $(10^3)^4$
2^{50}	pebi	Pi	petabinary: $(2^{10})^5$	peta: $(10^3)^5$
2^{60}	exbi	Ei	exabinary: $(2^{10})^6$	exa: $(10^3)^6$

kibi, mebi, gibi, tebi, pebi, exbi, zebi, yobi

en.wikipedia.org/wiki/Binary_prefix

- New IEC Standard Prefixes [only to exbi officially]

Name	Abbr	Factor
kibi	Ki	2^{10} = 1,024
mebi	Mi	2^{20} = 1,048,576
gibi	Gi	2^{30} = 1,073,741,824
tebi	Ti	2^{40} = 1,099,511,627,776
pebi	Pi	2^{50} = 1,125,899,906,842,624
exbi	Ei	2^{60} = 1,152,921,504,606,846,976
zebi	Zi	2^{70} = 1,180,591,620,717,411,303,424
yobi	Yi	2^{80} = 1,208,925,819,614,629,174,706,176

- International Electrotechnical Commission (IEC) in 1999 introduced these to specify binary quantities.
  - Names come from shortened versions of the original SI prefixes (same pronunciation) and *bi* is short for “binary”, but pronounced “bee” :-(
  - Now SI prefixes only have their base-10 meaning and never have a base-2 meaning.

kibi, mebi, gibi, tebi, pebi, exbi, zebi, yobi

en.wikipedia.org/wiki/Binary_prefix

- New IEC Standard Prefixes [only to exbi officially]

Name	Abbr	Factor
kibi	Ki	2^{10} = 1,024
mebi	Mi	2^{20} = 1,048,576
gibi	Gi	2^{30} = 1,073,741,824
tebi	Ti	2^{40} = 1,099,511,627,776
pebi	Pi	2^{50} = 1,125,899,906,842,624
exbi	Ei	2^{60} = 1,152,921,504,606,846,976
zebi	Zi	2^{70} = 1,180,591,620,717,411,303,424
yobi	Yi	2^{80} = 1,208,925,819,614,629,174,706,176

As of this writing, this proposal has yet to gain widespread use…

- International Electrotechnical Commission (IEC) in 1999 introduced these to specify binary quantities.
  - Names come from shortened versions of the original SI prefixes (same pronunciation) and *bi* is short for "binary", but pronounced "bee" :-(
  - Now SI prefixes only have their base-10 meaning and never have a base-2 meaning.

Memorization Help (Number Fluency)

- What is 2^{34}?

  x = 3

  y = 4

  $2^4 \times (2^{10})^3$

- What is 8 pebi?

  8 becomes y => 3

  pebi becomes x => 5

X=0 ⇒ ---	Y=0 ⇒ 1
X=1 ⇒ kibi ~10^3	Y=1 ⇒ 2
X=2 ⇒ mebi ~10^6	Y=2 ⇒ 4
X=3 ⇒ gibi ~10^9	Y=3 ⇒ 8
X=4 ⇒ tebi ~10^{12}	Y=4 ⇒ 16
X=5 ⇒ pebi ~10^{15}	Y=5 ⇒ 32
X=6 ⇒ exbi ~10^{18}	Y=6 ⇒ 64
X=7 ⇒ zebi ~10^{21}	Y=7 ⇒ 128
X=8 ⇒ yobi ~10^{24}	Y=8 ⇒ 256
	Y=9 ⇒ 512

- How many bits needed to address 2.5 Tebi ? ($2.5 \times (2^{10})^4$)
  - not a perfect multiple of 2, so round off
    - tebi is a 4: x => 4
    - 2.5 has 2 as the closest power of 2: y => 2
      - answer is 42 bits

Memorization Help (Number Fluency)

- What is 2^{34}?

  x = 3

  y = 4

  $2^4 \times (2^{10})^3$

- What is 8 pebi?

  8 becomes y => 3

  pebi becomes x => 5

$X=0 \Rightarrow$ ---	$Y=0 \Rightarrow 1$
$X=1 \Rightarrow$ kibi $\sim 10^3$	$Y=1 \Rightarrow 2$
$X=2 \Rightarrow$ mebi $\sim 10^6$	$Y=2 \Rightarrow 4$
$X=3 \Rightarrow$ gibi $\sim 10^9$	$Y=3 \Rightarrow 8$
$X=4 \Rightarrow$ tebi $\sim 10^{12}$	$Y=4 \Rightarrow 16$
$X=5 \Rightarrow$ pebi $\sim 10^{15}$	$Y=5 \Rightarrow 32$
$X=6 \Rightarrow$ exbi $\sim 10^{18}$	$Y=6 \Rightarrow 64$
$X=7 \Rightarrow$ zebi $\sim 10^{21}$	$Y=7 \Rightarrow 128$
$X=8 \Rightarrow$ yobi $\sim 10^{24}$	$Y=8 \Rightarrow 256$
	$Y=9 \Rightarrow 512$

- How many bits needed to address 2.5 Tebi ? $(2.5 \times (2^{10})^4)$