CS2100 COMPUTER ORGANISATION

http://www.comp.nus.edu.sg/~cs2100/

Lecture #3a

Data Representation and Number Systems

NUS National University of Singapore | School of Computing

Questions?

Ask at https://app.sli.do/event/qVCWNryB45Bnh6p2HRfnFG

OR

Scan and ask your questions here!
(May be obscured in some slides)

Lecture #3: Data Representation and Number Systems (1/2)

1. Data Representation
2. Decimal (base 10) Number System
3. Other Number Systems
4. Base-*R* to Decimal Conversion
5. Decimal to Binary Conversion
   - 5.1 Repeated Division-by-2
   - 5.2 Repeated Multiplication-by-2
6. Conversion Between Decimal and Other Bases
7. Conversion Between Bases
8. Binary to Octal/Hexadecimal Conversion

Lecture #3: Data Representation and Number Systems (2/2)

9. ASCII Code
10. Negative Numbers
    - 10.1 Sign-and-Magnitude
    - 10.2 1s Complement
    - 10.3 2s Complement
    - 10.4 Comparisons
    - 10.5 Complement on Fractions
    - 10.6 2s Complement Addition/Subtraction
    - 10.7 1s Complement Addition/Subtraction
    - 10.8 Excess Representation
11. Real Numbers
    - 11.1 Fixed-Point Representation
    - 11.2 Floating-Point Representation

1. Data Representation (1/2)

Basic data types in C:

int | float | double | char

Variants: short, long

How data is represented depends on its type:

01000110

As an 'int', it is 70

As a 'char', it is 'F'

11000000110100000000000000000000

As an 'int', it is -1060110336

As an 'float', it is -6.5

1. Data Representation (2/2)

- Data are internally represented as sequence of bits (***b***inary dig***it***s). A bit is either 0 or 1.
- Other units
  - Byte: 8 bits
  - Nibble: 4 bits (rarely used now)
  - Word: Multiple of bytes (eg: 1 byte, 2 bytes, 4 bytes, etc.) depending on the computer architecture
- N bits can represent up to 2^N values
  - Eg: 2 bits represent up to 4 values (00, 01, 10, 11);
    4 bits represent up to 16 values (0000, 0001, 0010, …., 1111)
- To represent M values, $\lceil \log_2 M \rceil$ bits required
  - Eg: 32 values require 5 bits; 1000 values require 10 bits

2. Decimal (base 10) Number System

- A weighted-positional number system.
- Base (also called radix) is 10
- Symbols/digits = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }
- Each position has a weight of power of 10
  - Eg: $(7594.36)_{10} = (7 \times 10^3) + (5 \times 10^2) + (9 \times 10^1) + (4 \times 10^0) + (3 \times 10^{-1}) + (6 \times 10^{-2})$

$$(a_n a_{n-1} \ldots a_0 \,.\, f_1 f_2 \ldots f_m)_{10} = (a_n \times 10^n) + (a_{n-1} \times 10^{n-1}) + \ldots + (a_0 \times 10^0) + (f_1 \times 10^{-1}) + (f_2 \times 10^{-2}) + \ldots + (f_m \times 10^{-m})$$

3. Other Number Systems (1/2)

- Binary (base 2)
  - Weights in powers of 2
  - Binary digits (bits): **0, 1**
- Octal (base 8)
  - Weights in powers of 8
  - Octal digits: **0, 1, 2, 3, 4, 5, 6, 7**.
- Hexadecimal (base 16)
  - Weights in powers of 16
  - Hexadecimal digits: **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F**.
- Base/radix R:
  - Weights in powers of R

3. Other Number Systems (2/2)

- In some programming languages/software, special notations are used to represent numbers in certain bases
  - In programming language C
    - Prefix 0 for octal. Eg: 032 represents the octal number $(32)_8$
    - Prefix 0x for hexadecimal. Eg: 0x32 represents the hexadecimal number $(32)_{16}$
  - In QTSpim (a MIPS simulator you will use)
    - Prefix 0x for hexadecimal. Eg: 0x100 represents the hexadecimal number $(100)_{16}$
  - In Verilog, the following values are the same
    - 8'b11110000: an 8-bit binary value 11110000
    - 8'hF0: an 8-bit binary value represented in hexadecimal F0
    - 8'd240: an 8-bit binary value represented in decimal 240

4. Base-*R* to Decimal Conversion

- Easy!
  - $1101.101_2 = 1\times2^3 + 1\times2^2 + 1\times2^0 + 1\times2^{-1} + 1\times2^{-3}$
    $= 8 + 4 + 1 + 0.5 + 0.125 = \mathbf{13.625_{10}}$
  - $572.6_8 = 5\times8^2 + 7\times8^1 + 2\times8^0 + 6\times8^{-1}$
    $= 320 + 56 + 2 + 0.75 = \mathbf{378.75_{10}}$
  - $2A.8_{16} = 2\times16^1 + 10\times16^0 + 8\times16^{-1}$
    $= 32 + 10 + 0.5 = \mathbf{42.5_{10}}$
  - $341.24_5 = 3\times5^2 + 4\times5^1 + 1\times5^0 + 2\times5^{-1} + 4\times5^{-2}$
    $= 75 + 20 + 1 + 0.4 + 0.16 = \mathbf{96.56_{10}}$
- DLD page 42 Quick Review Questions
  Questions 2-1 to 2-4.

5. Decimal to Binary Conversion

- For whole numbers
  - Repeated Division-by-2 Method
- For fractions
  - Repeated Multiplication-by-2 Method

5.1 Repeated Divison-by-2

- To convert a whole number to binary, use successive division by 2 until the quotient is 0. The remainders form the answer, with the first remainder as the *least significant bit (LSB)* and the last as the *most significant bit (MSB)*.

$(43)_{10} = ($ 101011 $)_2$

2	43		
2	21	rem 1	← LSB
2	10	rem 1	
2	5	rem 0	
2	2	rem 1	
2	1	rem 0	
	0	rem 1	← MSB

5.2 Repeated Multiplication-by-2

- To convert decimal fractions to binary, repeated multiplication by 2 is used, until the fractional product is 0 (or until the desired number of decimal places). The carried digits, or *carries*, produce the answer, with the first carry as the MSB, and the last as the LSB.

$(0.3125)_{10} = (\ .0101\ )_2$

	Carry	
$0.3125 \times 2 = 0.625$	0	←MSB
$0.625 \times 2 = 1.25$	1	
$0.25 \times 2 = 0.50$	0	
$0.5 \times 2 = 1.00$	1	←LSB

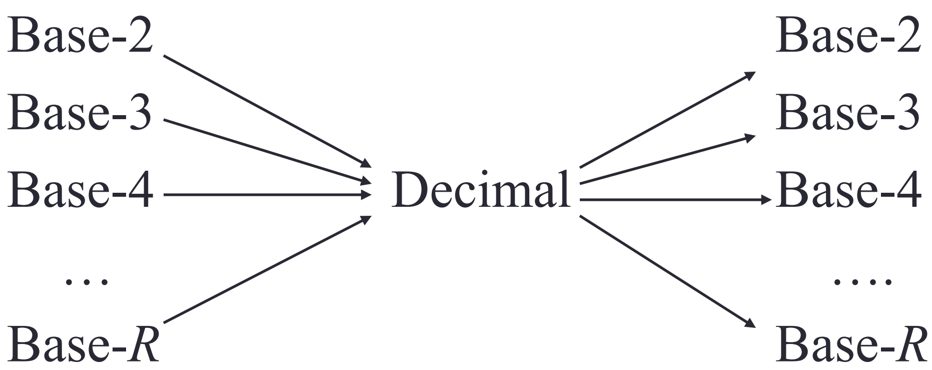
6. Conversion Between Decimal and Other Bases

- Base-*R* to decimal: multiply digits with their corresponding weights
- Decimal to binary (base 2)
  - Whole numbers: repeated division-by-2
  - Fractions: repeated multiplication-by-2
- Decimal to base-*R*
  - Whole numbers: repeated division-by-*R*
  - Fractions: repeated multiplication-by-*R*

- DLD page 42 Quick Review Questions Questions 2-5 to 2-8.

7. Conversion Between Bases

- In general, conversion between bases can be done via decimal:



- Shortcuts for conversion between bases 2, 4, 8, 16 (see next slide)

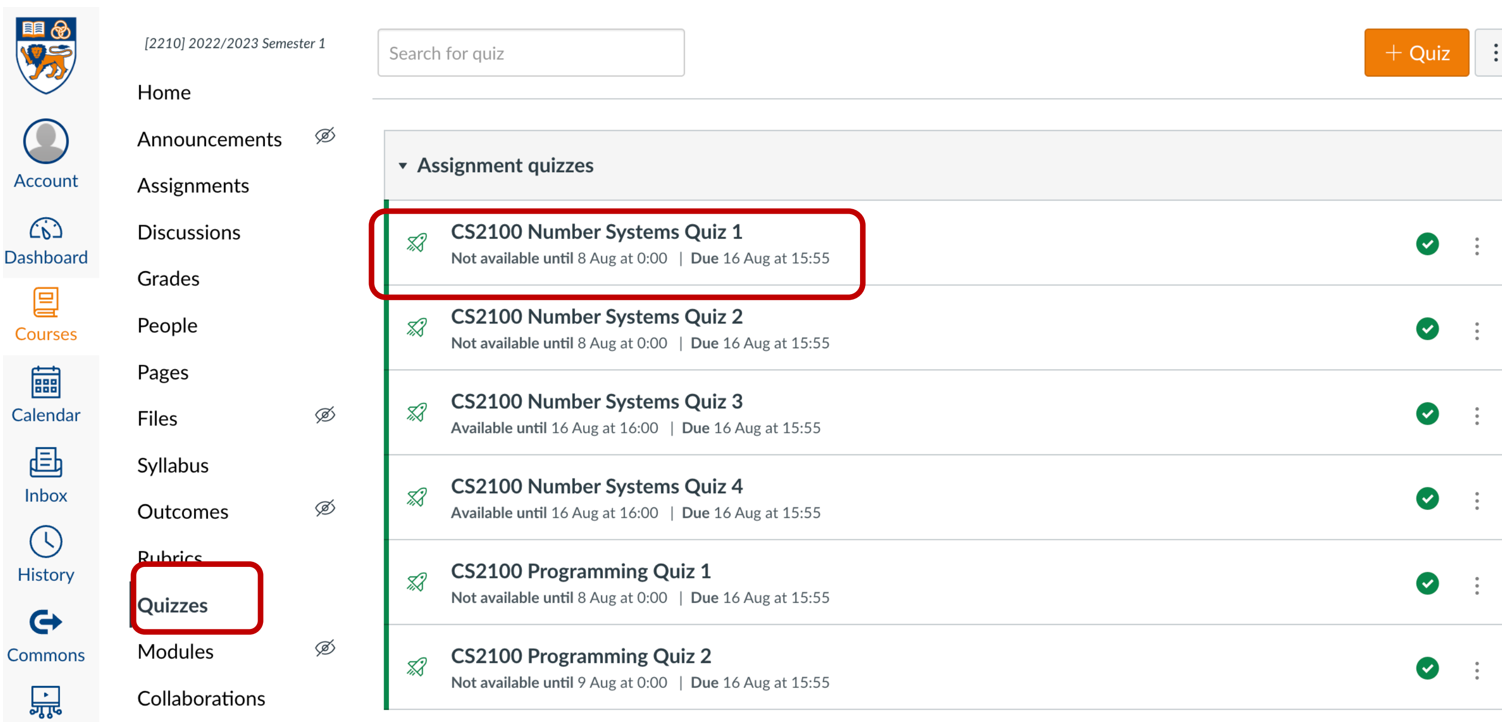
8. Binary to Octal/Hexadecimal Conversion

- Binary → Octal: partition in groups of 3
  - $(10\ 111\ 011\ 001\ .\ 101\ 110)_2$ = $\mathbf{(2731.56)_8}$
- Octal → Binary: reverse
  - $(2731.56)_8$ = $\mathbf{(10\ 111\ 011\ 001\ .\ 101\ 110)_2}$
- Binary → Hexadecimal: partition in groups of 4
  - $(101\ 1101\ 1001\ .\ 1011\ 1000)_2$ = $\mathbf{(5D9.B8)_{16}}$
- Hexadecimal → Binary: reverse
  - $(5D9.B8)_{16}$ = $\mathbf{(101\ 1101\ 1001\ .\ 1011\ 1000)_2}$

- DLD page 42 Quick Review Questions Questions 2-9 to 2-10.

Quiz

- Please complete the “CS2100 C Number Systems Quiz 1” in Canvas.
  - Access via the “Quizzes” tool in the left toolbar and select the quiz on the right side of the screen.



End of File