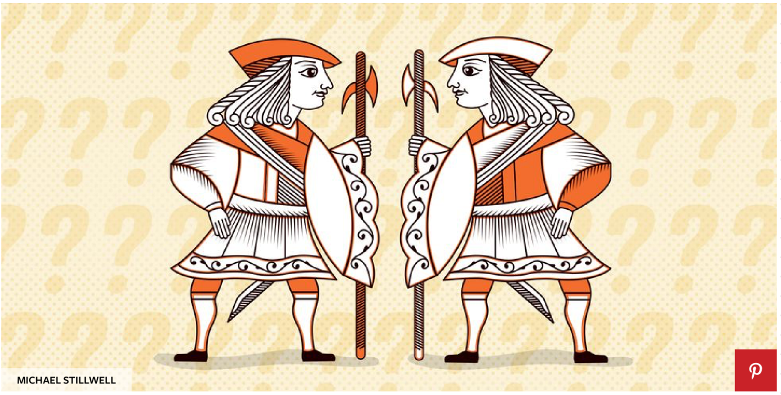


CSCI 2824: Discrete Structures

Lecture 3: Propositional Logic

Reminders

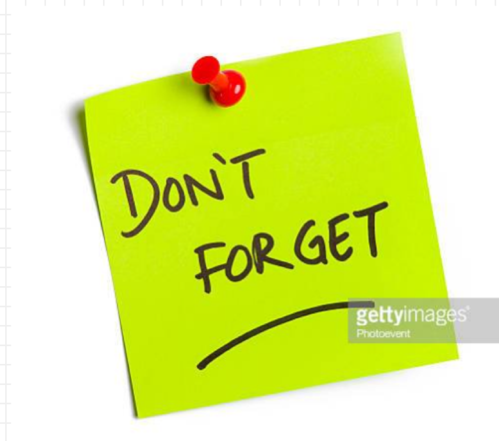
Submissions:

- Homework 1: Fri 9/6 at noon
- Quizlet 1: Wed 9/4 at 8 AM

Disabilities forms – turn them in by the end of 2nd week

Readings:

- 1.1-1.3 through next week



Example

Subtract $(11101)_2 - (110)_2$

$$
\begin{array}{r}
\rightarrow \quad 11101 \; - \\
\rightarrow \quad 110 \\
\hline
\rightarrow \quad 10111
\end{array}
\qquad\qquad
\begin{array}{r}
10111 \; + \\
110 \\
\hline
11101
\end{array}
$$

$$
\begin{array}{r}
-(10)_2 \\
01 \\
\hline
\end{array}
\quad
\begin{array}{l}
\longrightarrow \\
\longrightarrow
\end{array}
\quad
\begin{array}{r}
2_{10} \; - \\
1_{10} \\
\hline
1_{10}
\end{array}
\longrightarrow 1_2
$$

Subtracting two numbers in binary: Proceed right to left

In **decimal**: We can take 1 from the column to the left, and bring 10 to the right.

In **binary**: We can take 1 from the column to the left, and bring 2 to the right.

Warmup

How many bits are needed to encode all lowercase letters of the English alphabet?

2 bits 00, 01, 10, 11 $\to 4 = 2^2$

3 bits 000, 001, 010, 011, 100, 101, 110, 111 $\to 8 = 2^3$

n bits $\to 2^n$

a $\to$ 0

b $\to$ 1

- - -

z $\to$ 25

$(25)_{10} \to$ 11001

5-bits

$2^2 \to 4$

$2^3 \to 8$

$2^4 \to 16$

$2^5 \to 32$ ← 5-bits

$2^6 \to 64$

Intro to Python (Python 3)

Easiest way to get Python: https://www.anaconda.com/download

Good Practice: https://www.hackerrank.com/domains/python

More Examples IF WE HAVE TIME

Example

Convert 0.1 from decimal to binary

Powers of 2:

$$2^{-1} = \frac{1}{2}$$
$$2^{-2} = \frac{1}{4}$$
$$2^{-3} = \frac{1}{8}$$
$$2^{-4} = \frac{1}{16}$$
$$2^{-5} = \frac{1}{32}$$
$$2^{-6} = \frac{1}{64}$$
$$2^{-7} = \frac{1}{128}$$

$$\underset{2^7}{\_\_}\ \underset{2^6}{\_\_}\ \underset{2^5}{\_\_}\ \underset{2^4}{\_\_}\ \underset{2^3}{\_\_}\ \underset{2^2}{\_\_}\ \underset{2^1}{\_\_}\ \underset{2^0}{\_\_}\ \bullet\ \underset{2^{-1}}{\_\_}\ \underset{2^{-2}}{\_\_}\ \underset{2^{-3}}{\_\_}\ \underset{2^{-4}}{\_\_}\ \underset{2^{-5}}{\_\_}\ \underset{2^{-6}}{\_\_}\ \underset{2^{-7}}{\_\_}$$

Example

Powers of 2:

$$2^{-1} = \frac{1}{2}$$
$$2^{-2} = \frac{1}{4}$$
$$2^{-3} = \frac{1}{8}$$
$$2^{-4} = \frac{1}{16}$$
$$2^{-5} = \frac{1}{32}$$
$$2^{-6} = \frac{1}{64}$$
$$2^{-7} = \frac{1}{128}$$

Convert 0.1 from decimal to binary

Example: Convert 0.1 from decimal to binary.

0.1*2=0.2, so first bit is a 0

0.2*2=0.4, so next bit is a 0

0.4*2=0.8, so next bit is a 0

0.8*2=1.6, so next bit is a 1

0.6*2=1.2, so next bit is a 1

0.2*...NOW HOLD ON A SECOND!

- Restarts the pattern from here.

$0.1_{10} = 0.00011001100110011..._2$

								.	0	0	0	1	1		
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}

Example

$0.1_{10} = 0.00011001100110011..._{2}$

- So computers would need to store an infinite number of bits in order to store 0.1_{10} exactly.
- ... obviously, computers can't do that.
- They truncate at a certain point, leading to some error.
- **How bad can this error be?**

Powers of 2:

$$2^{-1} = \frac{1}{2}$$
$$2^{-2} = \frac{1}{4}$$
$$2^{-3} = \frac{1}{8}$$
$$2^{-4} = \frac{1}{16}$$
$$2^{-5} = \frac{1}{32}$$
$$2^{-6} = \frac{1}{64}$$
$$2^{-7} = \frac{1}{128}$$

2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0} . 2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-5} 2^{-6} 2^{-7}

Example: Truncation error.

$$f(a,b) = 333.75b^6 + a^2(11a^2b^2 - b^6 - 121b^4 - 2) + 5.5b^8 + \frac{a}{2b}$$

where $a = 77617, b = 33096$

$z = 333.75\mathrm{b}^6 + \mathrm{a}^2\,(11\mathrm{a}^2\,\mathrm{b}^2 - \mathrm{b}^6 - 121\mathrm{b}^4 - 2)$

$x = 5.5\mathrm{b}^8$

$y = z + x + a/(2b)$

$z = -7917111340668961361101134701524942850$

$x = 7917111340668961361101134701524942848$

$z + x = -2 \Longrightarrow y = -2 + a/(2b) = -0.827396\ldots$

But, if precision $p \le 35$, then

$z + x \cong 0 \quad \Longrightarrow y \cong \qquad (a/2b) = 1.1726\ldots$

Not even the correct sign!

Binary Representation System

Learning goals:

- Be able to convert between 2 bases (i.e., 10 and 2, 2 and 10)
- Be able to convert integers and also fractions
- Be able to add and subtract two numbers in base 2
- Be able to solve other problems using your knowledge of binary system
- *Show your work!*

Propositional Logic

Propositional logic: an introduction

- Gives precise meaning to mathematical statements.
- Statements have values of either **valid** or **invalid**.

- An important learning objective of this class is to understand and construct mathematical arguments.
- So we begin with **logic**.

Propositional Logic

Definition: The basic building block of logic is a **proposition**. A proposition is a declarative statement that is either true or false, but not both.

4 Types of Sentences

Declarative Sentence
- Tells something.
- Ends with a period. (.)

Interrogative Sentence
- Asks a question.
- Ends with a question mark. (?)

Exclamatory Sentence
- Shows strong feeling.
- Ends with a period. (!)

Imperative Sentence
- Gives a command.
- Ends with a period. (. or !)

Propositional Logic

Examples of Propositions

Boulder is a city in Colorado.

Golden is the capital of Colorado.

$2 + 2 = 5$

$1 + 2 = 3$

Examples of NOT Propositions

Don't do that.

Where are you going?

6

$x + 2 = 3$

Propositional Logic

Definition: The **truth value** of a proposition is **true** (denoted T) if the proposition is true; it is **false** (denoted F) if the proposition is false.

1. Boulder is a city in Colorado.

2. 2+2 = 5

Propositional Logic

Definition: Let p be a proposition. The **negation** of p, denoted by $\neg p$, is the proposition "It is not the case that p". The truth value of $\neg p$ is the opposite of the truth value of p.

Example: Let p denote the proposition: "It is raining today."

Then, $\neg \boldsymbol{p}$ denotes: "It is **not the case that** it is raining today." or "It is **not** raining today."

Propositional Logic

Definition: It is convenient to tabulate of the possible truth values for the various configurations of the propositions. This is done using a **truth table**.

Given simple propositions p and q, the truth table allows us to enumerate all possible truth values of combinations of p and q.

Example: Give the truth table for p and $\neg p$

p	$\neg p$

Propositional Logic – Connectives, Logical Operations

We can start combining propositions. We do this using logical operators called **<u>connectives</u>**.

Some examples:

- **conjunction**: “and”, denoted $\land$
- **disjunction**: “or”, denoted $\lor$
- **conditional**: “if-then”, denoted $\rightarrow$
- **biconditional**: “if and only if”, denoted $\Leftrightarrow$

Propositional Logic – Connectives, Logical Operations

Definition: Let *p* and *q* be propositions. The **conjunction** of *p* and *q*, denoted by $p \wedge q$, is the proposition "*p* and *q*". The conjunction $p \wedge q$ has the truth value T if both *p* and *q* are T and is F otherwise.

<u>Example:</u> Let p = "it is dark outside" and q = "my house is haunted"

"It is dark outside **and** my house is haunted"

"It is light outside, but my house is still haunted"

Propositional Logic – Connectives, Logical Operations

Definition: Let p and q be propositions. The **conjunction** of p and q, denoted by $p \wedge q$, is the proposition "p and q". The conjunction $p \wedge q$ has the truth value T if both p and q are T and is F otherwise.

Example: Let p = "it is dark outside" and q = "my house is haunted"

"It is dark outside and my house is haunted" … is the conjunction $p \wedge q$

"It is light outside, but my house is still haunted" … is the conjunction $\neg p \wedge q$

Truth Table for a Conjunction: $p \wedge q$

p	q	$p \wedge q$

Truth Table for a Conjunction: $p \land q$

p	q	$p \land q$
T	T	T
T	F	F
F	T	F
F	F	F

Propositional Logic – Connectives, Logical Operations

Definition: Let *p* and *q* be propositions. The **disjunction** of *p* and *q*, denoted by $p \vee q$, is the proposition "*p* or *q*". The disjunction $p \vee q$ has the truth value T if either *p* or *q* are T and is F otherwise.

Example: Let p = "it is dark outside" and q = "my house is haunted"

It is dark out side ***or*** my house is haunted.

This is an "inclusive or". So one proposition could be true, or they both could be true and the overall statement would be true.

Truth Table for a Disjunction: $p \vee q$

p	q	$p \vee q$

Truth Table for a Disjunction: $p \lor q$

p	q	$p \lor q$
T	T	T
T	F	T
F	T	T
F	F	F

Propositional Logic – Connectives, Logical Operations

Definition: Let *p* and *q* be propositions. The **exclusive or** of *p* and *q*, denoted $p \oplus q$, is the proposition that is true when exactly one of *p* or *q* is true, and false otherwise.

- Also abbreviated as "xor" sometimes

Example: Let p = "It is daytime." and q = "It is nighttime."

It is daytime ***or*** it is nighttime.

"Exclusive or" means that we could have either one of the propositions be true, **but not both**.

Truth Table for an Exclusive Or: $p \oplus q$

p	q	$p \oplus q$

Truth Table for an Exclusive Or: $p \oplus q$

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

Propositional Logic – Connectives, Logical Operations

Definition: Let p and q be two propositions. The **conditional** "if p then q", denoted by $p \rightarrow q$, is false when p is true but q is false, and true otherwise.

- The conditional describes an *if-then* relationship between the two propositions.
- Think of the conditional $p \rightarrow q$ as defining a rule. What are the cases where the rule holds or where the rule is broken.

<u>Example:</u> If you get a 100% on the final, then you will get an A.

Other ways to express $p \longrightarrow q$:

If p, then q.
If p, q.
p is sufficient for q.
q if p.
q when p.
q unless $\neg p$.
p implies q
p only if q
A sufficient condition for q is p.
q whenever p
q is necessary for p
q follows from p

Truth Table for a Conditional: $p \longrightarrow q$

p	q	$p \longrightarrow q$

Truth Table for a Conditional: $p \longrightarrow q$

p	q	$p \longrightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Example: “If I go for a run, then I am happy.”

p = “I go for a run”

q = “I am happy”

In symbols: $p \rightarrow q$ if [I go for a run], then [I am happy]

How about a truth table for $p \rightarrow q$?

- p=T, q=T: *I go for a run* and *I am happy*.
    - Consistent with our rule, so $p \rightarrow q$ is **true**.
- p=F, q=F: *I didn’t go for a run* and *I am not happy*.
    - Rule is not broken, so $p \rightarrow q$ is **true**.
- p=F, q=T: *I didn’t go for a run* and *I am happy.*
    - Rule is not broken (***why?***), so $p \rightarrow q$ is **true**.
- p=T, q=F: *I go for a run* and *I am not happy*.
    - If going for a run implies that I should be happy, then this is **not consistent** with our rule, so $p \rightarrow q$ is **false**.

Example: Translate the following plain English conditionals into symbols.

(a) *If it snows, then I crash my bicycle.*

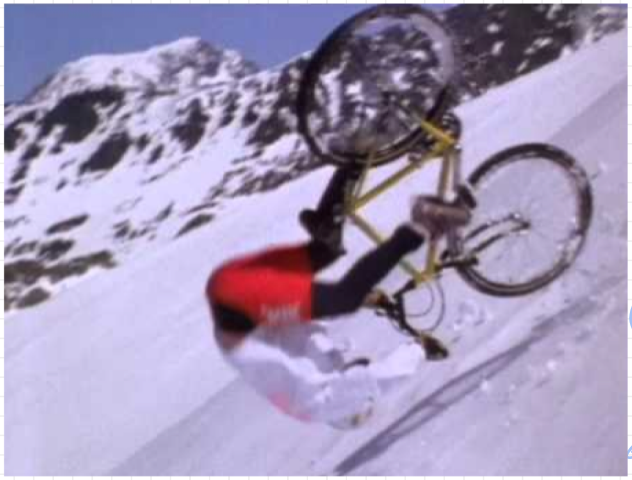
Example: Translate the following plain English conditionals into symbols.

(a) *If it snows, then I crash my bicycle.*

Define: p = “it snows” and q = “I crash my bike”

and this statement can be expressed as $p \rightarrow q$.

(a) What about: *I crash my bicycle only if it snows.*



Example: Translate the following plain English conditionals into symbols.

(a) *If it snows, then I crash my bicycle.*

Define: p = "it snows" and q = "I crash my bike"

and this statement can be expressed as $p \rightarrow q$.

(a) What about: *I crash my bicycle only if it snows.*

Caution! Which of these events ensures the other?

Example: Translat

(a) *If it sno*

Define: p = "

and this stat

Note: Just because the "if" was attached to *it snows* does not mean that's the "if" part of the conditional.

Even though *causality* might work that direction (the snow caused me to crash), *logic/information* does not (you know it's snowing *because* I crashed).

(a) What about: *I crash my bicycle only if it snows.*

Caution! Which of these events ensures the other?

$q \rightarrow p$

Propositional Logic – Connectives, Logical Operations

Definition: Let *p* and *q* be two propositions. The **biconditional** "*p* if and only if *q*", denoted by $p \Leftrightarrow q$, or *p* **iff** *q*, is true when *p* and *q* have the same truth value, and false otherwise.

- The conditional describes an *if-and-only-if* relationship between the two propositions.

Example: Let p = "A polygon has 3 sides." and q = "It is a triangle."

A polygon has 3 sides **if and only if** it is a triangle

Truth Table for a biconditional: $p \iff q$

p	q	$p \iff q$

Truth Table for a biconditional: $p \iff q$

p	q	$p \iff q$
T	T	T
T	F	F
F	T	F
F	F	T

Propositional Logic – Compound Propositions

Compound propositions are constructed by linking together multiple simple propositions using connectives.

Example: Determine the truth table for the compound proposition

$$(p \rightarrow q) \wedge (q \rightarrow p)$$

For example: "If I think I see a ghost, then I am superstitious."

and "If I am superstitious, then I think I see ghosts."

Propositional Logic – Compound Propositions

Compound propositions are constructed by linking together multiple simple propositions using connectives.

Order of Operations/Precedence of Logical Operators

1. Negation
2. Conjunction over disjunction $p \land q \lor r$ means $(p \land q) \lor r$
3. Conditionals, biconditionals $p \lor q \longrightarrow r$ means $(p \lor q) \longrightarrow r$

Propositional Logic – Compound Propositions

Example: Construct a truth table for $(p \longrightarrow q) \land (\neg p \longrightarrow r)$

p	**q**	**r**	$\neg p$	$p \longrightarrow q$	$\neg p \longrightarrow r$	$(p \longrightarrow q) \land (\neg p \longrightarrow r)$

Propositional Logic – Compound Propositions

Compound propositions are constructed by linking together multiple simple propositions using connectives.

Example: Determine the truth table for the compound proposition $(p \rightarrow q) \land (q \rightarrow p)$

p	q	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \land (q \rightarrow p)$
T	T			
T	F			
F	T			
F	F			

Propositional Logic – Compound Propositions

Compound propositions are constructed by linking together multiple simple propositions using connectives.

Example: Determine the truth table for the compound proposition $(p \rightarrow q) \wedge (q \rightarrow p)$

p	q	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

Propositional Logic – Compound Propositions

Does the truth table for $(p \rightarrow q) \land (q \rightarrow p)$ look familiar?

p	q	$(p \rightarrow q) \land (q \rightarrow p)$
T	T	T
T	F	F
F	T	F
F	F	T

Propositional Logic – Compound Propositions

Does the truth table for $(p \rightarrow q) \wedge (q \rightarrow p)$ look familiar?

… maybe to the biconditional, $p \Leftrightarrow q$?

p	q	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T
T	F	F
F	T	F
F	F	T

p	q	$p \Leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Because the two propositions have the same truth values for all possible values of p and q, they are **logically equivalent**. (more on that later)

Propositional Logic – Knights and Knaves

Example: The island of Knights and Knaves. Suppose you are on an island where there are two types of people: Knights always tell the truth, and Knaves always lie.

Suppose on this island, you encounter two people, Alfred and Batman. Let's call them A and B for short. Suppose A tells you "I am a Knave or B is a Knight." Use a truth table to determine what kind of people A and B are.

p: Alfred is a Knight.

q: Batman is a Knight.

A's statement: $\neg p \vee q$

Propositional Logic – Knights and Knaves

$$\boldsymbol{p} \Leftrightarrow (\neg \boldsymbol{p} \vee \boldsymbol{q})$$

It must be True on this island that if A is a Knight, then his statement is true and if A's statement is true, then he must be a knight.

p	q			
T	T			
T	F			
F	T			
F	F			

Propositional logic

FYOG: Suppose instead that *A* tells you "*B* is a Knight" and *B* tells you "The two of us are of different types". Use a truth table to determine the sorts of people that *A* and *B* are. (aside from confusing)

Let p = "A is a knight" and let q = "B is a knight"

Then A's statement is q, which holds if and only if A is telling the truth: $p \Leftrightarrow q$

and B's statement can be represented in a few different ways.

- The simplest way is probably: $(p \land \neg q) \lor (\neg p \land q)$
- The most compact way is with the exclusive-or: $p \oplus q$

So to represent the fact that B's statement is true **if and only if** B is a knight (i.e., telling the truth), we have: $q \Leftrightarrow p \oplus q$

Now, *both* A and B have made statements, and we want to include *both* of them in our truth table test. Since we want to test both, we link them together with a conjunction:

FYOG (continued): Suppose instead that A tells you "B is a Knight" and B tells you "The two of us are of different types". Use a truth table to determine the sorts of people that A and B are.

Now, *both* A and B have made statements, and we want to include *both* of them in our truth table test. Since we want to test both, we link them together with a conjunction:

→ Test when this compound proposition is true: $(p \Leftrightarrow q) \wedge (q \Leftrightarrow p \oplus q)$

p	q	$p \Leftrightarrow q$	$p \oplus q$	$q \Leftrightarrow p \oplus q$	$(p \Leftrightarrow q) \wedge (q \Leftrightarrow p \oplus q)$
T	T	T	F	F	F
T	F	F	T	F	F
F	T	F	T	T	F
F	F	T	F	T	T

The only row where our compound proposition is T is the last one, so both A and B must be knaves.

Floating Point Number Representation

- If x is a real number then its normal form representation is:

  $x = f \bullet \text{Base}^{E}$

  where f : mantissa, E: exponent

exponent

Example: $125.32_{10} = 0.12532 \bullet 10^{3}$

mantissa

$- 125.32_{10} = - 0.12532 \bullet 10^{3}$

$0.0546_{10} = 0.546 \bullet 10^{-1}$

- The mantissa is normalized, so the digit after the fractional point is non-zero. Note that in binary, the leading digit is always 1, so it is normally *hidden*.
- If needed the mantissa should be shifted appropriately to make the first digit (after the fractional point) to be non-zero & the exponent is properly adjusted.

Courtesy of Dale Rosen IUPUI

Normalizing Numbers

<u>Example</u>:

$134.15_{10} = 0.13415 \times 10^{3}$

$0.0021_{10} = 0.21 \times 10^{-2}$

$101.11_B = .1011 \times 2^{3}$ or 1.011×2^{2} (hidden1)

$0.011_B = .11 \times 2^{-1}$ or 1.1×2^{-2} (hidden1)

$AB.CD_H = .ABCD \times 16^{2}$

$0.00AC_H = .AC \times 16^{-2}$

- Note that the concept of a hidden 1 only applied to binary.

Floating Point Number Representation

Assume we use 16-bit binary pattern for normalized binary form based on the following convention (MSB to LSB)

- *Sign of mantissa* (±)= left most bit (where 0: +; 1: -)
- *Mantissa* (*f*)= next 11 bits, leading 1 is assumed, m=1.*f*
- *Exponent (E)* = next 4 bits, bias 7
  - 2^0=7 (0111). 2^1=8 (1000), 2^{-1}=6 (0110)

$$x = \pm f \bullet \text{Base}^{E}$$

$f = 1.?_1?_2?_3?_4\ldots?_{11}\,?_{12}\ldots?_{15}$

E : converted to binary, $b_1b_2b_3b_4$

MSB ... LSB

	$?_1$	$?_2$	$?_3$	$?_4$	$?_5$	$?_6$	$?_7$	$?_8$	$?_9$	$?_{10}$	$?_{11}$	b_1	b_2	b_3	b_4

+ : 0
- : 1

Excess-7

Floating Point Number Representation

Question:

How the computer expresses the 16-bit approximation of 1110.111010111111 in normalized binary form using the following convention

Sign of mantissa = left most bit (where 0: +; 1: -)
Mantissa = next 11 bits, leading 1 is hidden, really represents 12 bits
Exponent = next four bits, bias 7

sign	mantissa	exponent
1 bit	11 bits	4 bits

Answer:

Step 1: Normalization

1110.111010111111 = + 1.110111010111111 * 2 $^{+3}$

Step 2: "Plant" 16 bits

the 16 bit floating point representation is 0 11011101011 1010

*3 + **7** = 10 -> 1010_2*

Floating Point Number Representation

Question:

Interpret the normalized binary number

$0110\ 0000\ 0000\ 0100_B$

using the convention mentioned

Sign of mantissa = left most bit (where 0: +; 1: -)

Mantissa = next 11 bits, leading 1 is hidden, really represents 12 bits

Exponent = next four bits, bias 7

find its decimal equivalent.

Answer:

$\underline{0}\ \underline{11000000000}\ \underline{0100}_B = 1.11_B * 2^{(4-7)=-3} = 0.00111_B$

$= 7/32 = 0.21875_D$

Real Life Example: IEEE 754

- IEEE Standard 754 is the representation of floating point used on most computers.
- Single precision (float) is 32 bits or 4 bytes with the following configuration.

1 sign bit	8 exponent	23 fraction

•The sign field for mantissa is 0 for positive or 1 for negative

•In the mantissa, the decimal point is assumed to follow the first ‘1’. Since the first digit is always a ‘1’, a hidden bit is used to representing the bit. The fraction is the 23 bits following the first ‘1’. The fraction really represents a 24 bit mantissa.

•The exponent field has a bias of 127, meaning that 127 is added to the exponent before it’s stored. 20 becomes 127, 21 becomes 128, 2-3 becomes 124, 2-1 becomes 126, etc. When the exponent becomes -127 (all zeroes), the hidden bit is not used to allow gradual underflow.

Real Life Example: IEEE 754

IEEE 754 Examples: Normalized Numbers

0 1000 0011 0000 0000 0000 0000 0000 000

$= 1 \times 2^{4} = 16$

0 0011 0001 0000 0000 0000 0000 0000 000

$= 1 \times 2^{-78} = 3.3087e-24$

0 1000 0001 0100 0000 0000 0000 0000 000

$= 1.25 \times 2^{2} = 5$

Real Life Example: IEEE 754

Double precision (double) is 64 bites or 8 bytes with the following configuration.

1 sign bit	11 exponent	52 fraction

- The definition of the fields matches single precision.
- The double precision bias is 1023.
- What value can you not represent because of the hidden bit?
- Certain bit patterns are reserved to represent special values. Of particular importance is the representation for zero (all bits zero). There are also patterns to represent infinity, positive and negative numeric overflow, positive and negative numeric underflow, and not-a-number (abbreviated NaN).