Computer Organization and Assembly Language (COAL)

Lecture 2

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

- Data Representation
- Boolean Operations

Binary Numbers

- Digits are 1 and 0
  - 1 = true   In physical terms some voltage exists (2V - 5 V)
  - 0 = false In physical terms no voltage exists (0V – 1V)

- MSB – most significant bit
- LSB – least significant bit
- Bit numbering:

MSB   LSB

1 0 1 1 0 0 1 0 1 0 0 1 1 1 0 0

15   0

Binary Numbers

- Each bit represents a power of 2:

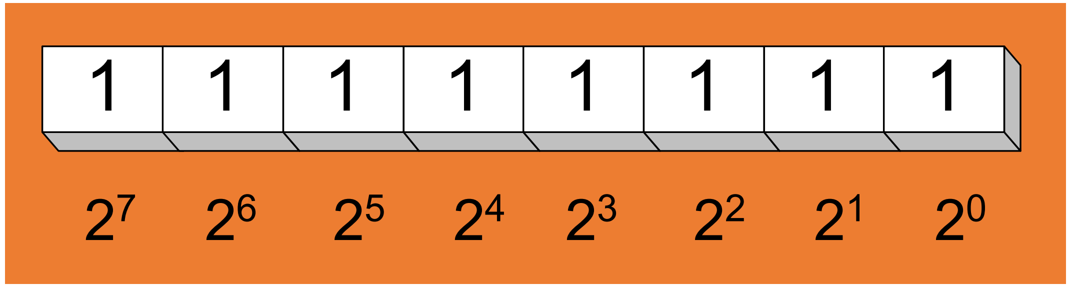


Table 1-3 Binary Bit Position Values.

2^n	Decimal Value	2^n	Decimal Value
2^0	1	2^8	256
2^1	2	2^9	512
2^2	4	2^{10}	1024
2^3	8	2^{11}	2048
2^4	16	2^{12}	4096
2^5	32	2^{13}	8192
2^6	64	2^{14}	16384
2^7	128	2^{15}	32768

Every binary number is a sum of powers of 2

Converting Binary to Decimal

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

$$Decimal = (d_{n-1} \times 2^{n-1}) + (d_{n-2} \times 2^{n-2}) + \ldots + (d_1 \times 2^1) + (d_0 \times 2^0)$$

d = binary digit

binary 00001001 = decimal 9:

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

Convert Unsigned Decimal to Binary

- Repeatedly divide the decimal integer by 2. Each remainder is a binary digit in the translated value. Example of “**37**”

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

Convert Unsigned Decimal to Binary

- 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 / 2		
4 / 2		
2 / 2		
1 / 2		

Convert Unsigned Decimal to Binary

- 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 / 2		
2 / 2		
1 / 2		

Convert Unsigned Decimal to Binary

- 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 / 2		
1 / 2		

Convert Unsigned Decimal to Binary

- 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 / 2		

Convert Unsigned Decimal to Binary

- 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		

Convert Unsigned Decimal to Binary

- 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

Convert Unsigned Decimal to Binary

- 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

least significant bit

most significant bit

stop when quotient is zero

37 = 100101

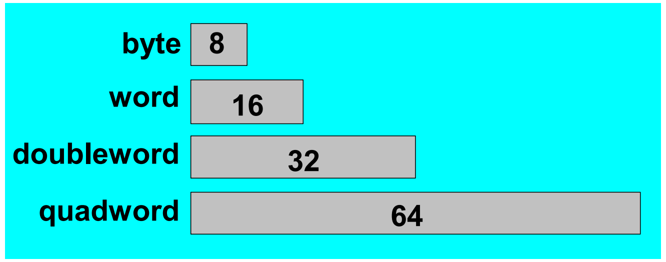
Binary Addition

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

carry:					1				
	0	0	0	0	0	1	0	0	(4)
+	0	0	0	0	0	1	1	1	(7)
	0	0	0	0	1	0	1	1	(11)
bit position:	7	6	5	4	3	2	1	0	

Integer Storage Sizes

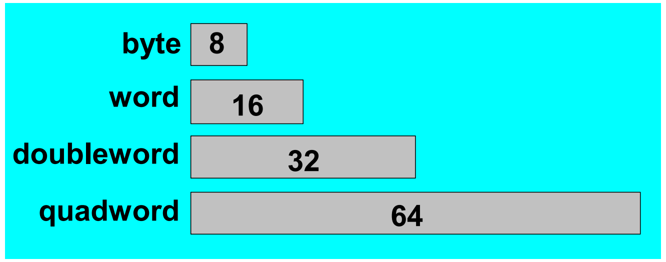
Standard sizes:



Storage Type	Range (low–high)	Powers of 2
Unsigned byte		
Unsigned word		
Unsigned doubleword		
Unsigned quadword		

Integer Storage Sizes

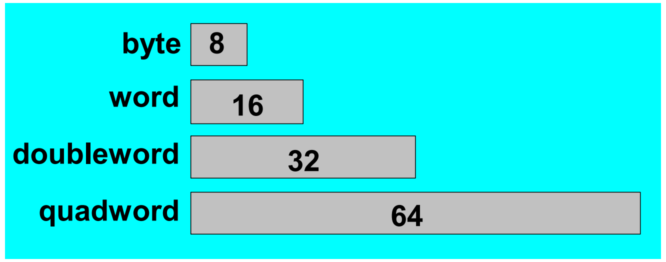
Standard sizes:



Storage Type	Range (low–high)	Powers of 2
Unsigned byte	0 to 255	0 to $(2^8 - 1)$
Unsigned word		
Unsigned doubleword		
Unsigned quadword		

Integer Storage Sizes

Standard sizes:



Storage Type	Range (low–high)	Powers of 2
Unsigned byte	0 to 255	0 to $(2^8 - 1)$
Unsigned word	0 to 65,535	0 to $(2^{16} - 1)$
Unsigned doubleword	0 to 4,294,967,295	0 to $(2^{32} - 1)$
Unsigned quadword	0 to 18,446,744,073,709,551,615	0 to $(2^{64} - 1)$

What is the largest unsigned integer that may be stored in **20 bits**?

Hexadecimal Integers

Binary values are represented in hexadecimal.

Binary	Decimal	Hexadecimal	Binary	Decimal	Hexadecimal
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	10	A
0011	3	3	1011	11	B
0100	4	4	1100	12	C
0101	5	5	1101	13	D
0110	6	6	1110	14	E
0111	7	7	1111	15	F

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer
  000101101010011110010100 to hexadecimal:

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001					

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001	0110				

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001	0110	1010			

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001	0110	1010	0111		

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001	0110	1010	0111	1001	

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1					
0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1	6				
0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1	6	A			
0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1	6	A	7		
0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1	6	A	7	9	
0001	0110	1010	0111	1001	0100

Translating Binary to Hexadecimal

- Each hexadecimal digit corresponds to 4 binary bits.
- Example: Translate the binary integer **000101101010011110010100** to hexadecimal:

1	6	A	7	9	4
0001	0110	1010	0111	1001	0100

Converting Hexadecimal to Decimal

- Multiply each digit by its corresponding power of 16:

  $$\text{dec} = (D_3 \times 16^3) + (D_2 \times 16^2) + (D_1 \times 16^1) + (D_0 \times 16^0)$$

- Hex **1234** equals $(1 \times 16^3) + (2 \times 16^2) + (3 \times 16^1) + (4 \times 16^0)$, or decimal **4,660**.

- Hex **3BA4** equals $(3 \times 16^3) + (11 * 16^2) + (10 \times 16^1) + (4 \times 16^0)$, or decimal **15,268**.

Powers of 16

Used when calculating hexadecimal values up to 8 digits long:

16^n	Decimal Value	16^n	Decimal Value
16^0	1	16^4	65,536
16^1	16	16^5	1,048,576
16^2	256	16^6	16,777,216
16^3	4096	16^7	268,435,456

Converting Decimal to Hexadecimal

Division	Quotient	Remainder
422 / 16	26	6
26 / 16	1	A
1 / 16	0	1

decimal 422 = 1A6 hexadecimal

Hexadecimal Addition

- Divide the sum of two digits by the number base (16). The quotient becomes the carry value, and the remainder is the sum digit.

		1	1
36	28	28	6A
42	45	58	4B
78	6D	80	B5

21 / 16 = 1, rem 5

Important skill: Programmers frequently add and subtract the addresses of variables and instructions.

Hexadecimal Subtraction

- When a borrow is required from the digit to the left, add 16 (decimal) to the current digit's value:

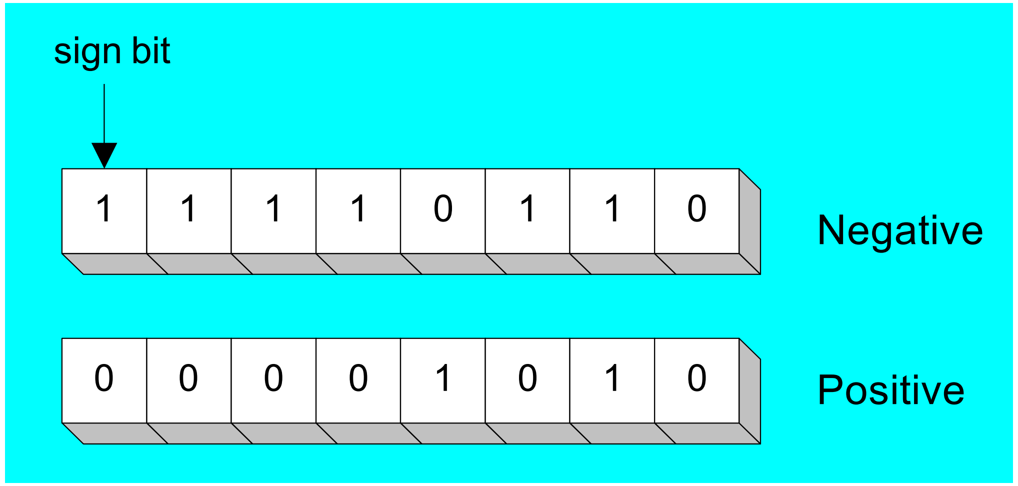
$\boxed{16 + 5 = 21}$

$$
\begin{array}{rr}
 & -1 \\
C6 & 75 \\
A2 & 47 \\
\hline
24 & 2E
\end{array}
$$

Practice: The address of **var1** is 00400020. The address of the next variable after var1 is 0040006A. **How many bytes are used by var1**?

Signed Integers

The highest bit indicates the sign. 1 = negative, 0 = positive



If the highest digit of a hexadecimal integer is > 7, the value is negative. Examples: 8A, C5, A2, 9D

Forming the Two's Complement

- Negative numbers are stored in two's complement notation
- Represents the **additive Inverse**

Starting value	00000001
Step 1: reverse the bits	11111110
Step 2: add 1 to the value from Step 1	11111110 +00000001
Sum: two's complement representation	11111111

Note that 00000001 + 11111111 = 00000000

Binary Subtraction

- When subtracting A – B, convert B to its two's complement
- Add A to (–B)

$$\begin{array}{r} 0\,0\,0\,0\,1\,1\,0\,0 \\ -\ 0\,0\,0\,0\,0\,0\,1\,1 \\ \hline \end{array} \longrightarrow \begin{array}{r} 0\,0\,0\,0\,1\,1\,0\,0 \\ 1\,1\,1\,1\,1\,1\,0\,1 \\ \hline 0\,0\,0\,0\,1\,0\,0\,1 \end{array}$$

Practice: Subtract 0101 from 1001.

Hexadecimal Two’s Complement

- Reverse all bits and add 1
- An easy way to reverse the bits of a hexadecimal digit is to subtract the digit from 15

```
6A3D --> 95C2 + 1 --> 95C3
95C3 --> 6A3C + 1 --> 6A3D
```

Translating Signed Binary to Decimal

- If the highest bit is a 1, the number is stored in two's-complement notation
- Take two's-complement again and convert this new number to decimal as if it were an unsigned binary integer.
- If the highest bit is a 0, you can convert it to decimal as if it were an unsigned binary integer

Starting value	11110000
Step 1: Reverse the bits	00001111
Step 2: Add 1 to the value from Step 1	00001111 + 1
Step 3: Create the two's complement	00010000
Step 4: Convert to decimal	16

Translating Signed Decimal to Binary

- Convert the absolute value of the decimal integer to binary
- If the original decimal integer was negative, create the two's complement of the binary number from the previous step

Translating Signed Decimal to Hexadecimal

- Convert the absolute value of the decimal integer to hexadecimal
- If the decimal integer was negative, create the two's complement of the hexadecimal number from the previous step.

Translating Signed Hexadecimal to Decimal

- If the hexadecimal integer is negative, create its two's complement; otherwise, retain the integer as is
- Using the integer from the previous step, convert it to decimal. If the original value was negative, attach a minus sign to the beginning of the decimal integer.

Ranges of Signed Integers

The highest bit is reserved for the sign. This limits the range:

Storage Type	Range (low–high)	Powers of 2
Signed byte	–128 to +127	-2^7 to $(2^7 - 1)$
Signed word	–32,768 to +32,767	-2^{15} to $(2^{15} - 1)$
Signed doubleword	–2,147,483,648 to 2,147,483,647	-2^{31} to $(2^{31} - 1)$
Signed quadword	–9,223,372,036,854,775,808 to +9,223,372,036,854,775,807	-2^{63} to $(2^{63} - 1)$

Practice: What is the largest positive value that may be stored in **20 bits**?

Character Storage

- Character sets
  - Standard ASCII (0 – 127)
  - Extended ASCII (0 – 255)
  - ANSI (0 – 255)
  - Unicode (0 – 65,535)
- Null-terminated String
  - Array of characters followed by a *null byte*
- Using the ASCII table
  - back inside cover of book

Reading Assignment

Read Character sets from book including the interpretation of ASCII table and ASCII control characters

Numeric Data Representation

- pure binary
  - can be calculated directly
- ASCII binary
  - string of digits: "01010101"
- ASCII decimal
  - string of digits: "65"
- ASCII hexadecimal
  - string of digits: "9C"

Boolean Operations

Boolean Operations

- NOT
- AND
- OR
- Operator Precedence
- Truth Tables

Boolean Algebra

- Based on **symbolic logic**, designed by George Boole
- Boolean expressions created from:
  - NOT, AND, OR

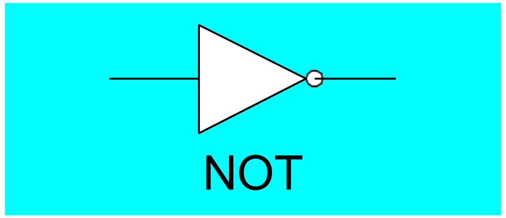
Expression	Description
$\neg X$	NOT X
$X \wedge Y$	X AND Y
$X \vee Y$	X OR Y
$\neg X \vee Y$	(NOT X) OR Y
$\neg (X \wedge Y)$	NOT (X AND Y)
$X \wedge \neg Y$	X AND (NOT Y)

NOT

- Inverts (reverses) a boolean value
- Truth table for Boolean NOT operator:

X	¬X
F	T
T	F

Digital gate diagram for NOT:

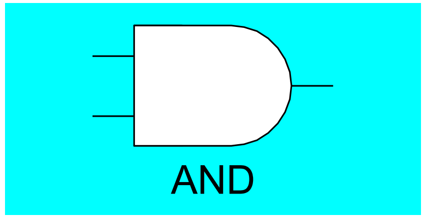


AND

- Truth table for Boolean AND operator:

X	Y	X ∧ Y
F	F	F
F	T	F
T	F	F
T	T	T

Digital gate diagram for AND:

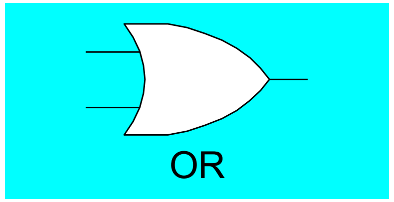


OR

- Truth table for Boolean OR operator:

X	Y	$X \vee Y$
F	F	F
F	T	T
T	F	T
T	T	T

Digital gate diagram for OR:



Operator Precedence

- Examples showing the order of operations:

Expression	Order of Operations
$\neg X \vee Y$	NOT, then OR
$\neg(X \vee Y)$	OR, then NOT
$X \vee (Y \wedge Z)$	AND, then OR

Truth Tables (1 of 3)

- A **Boolean function** has one or more Boolean inputs, and returns a single Boolean output.
- A **truth table** shows all the inputs and outputs of a Boolean function

X	¬X	Y	¬X ∨ Y
F	T	F	T
F	T	T	T
T	F	F	F
T	F	T	T

Truth Tables (2 of 3)

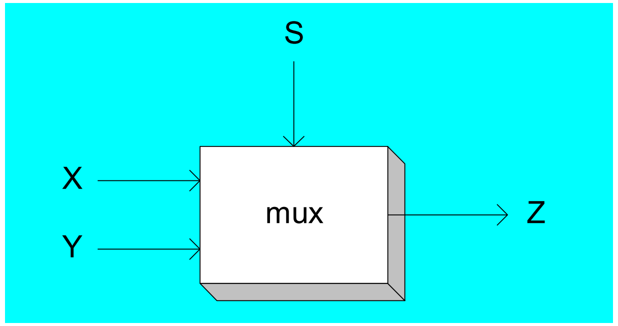
- Example: $X \wedge \neg Y$

X	Y	$\neg$Y	X $\wedge \neg$Y
F	F	T	F
F	T	F	F
T	F	T	T
T	T	F	F

Truth Tables (3 of 3)

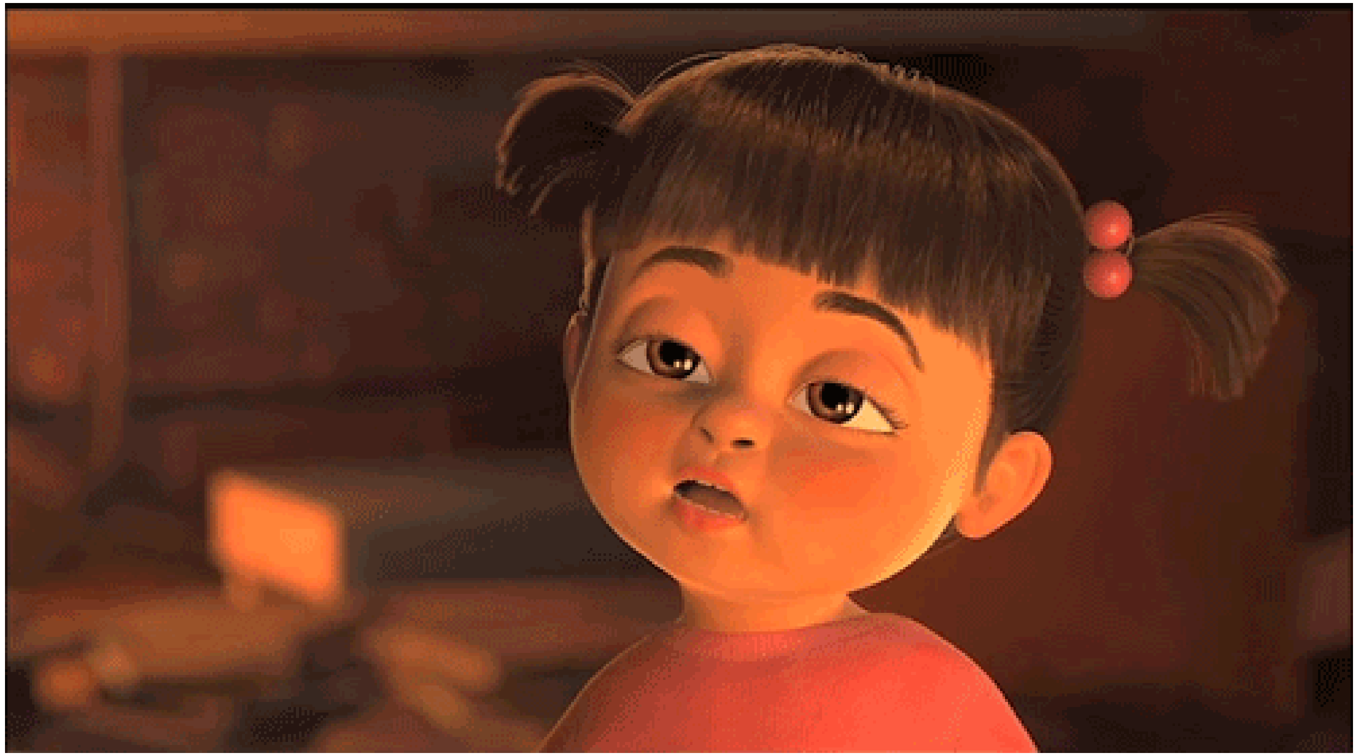
- Example: $(Y \wedge S) \vee (X \wedge \neg S)$

X	Y	S	$Y \wedge S$	$\neg S$	$X \wedge \neg S$	$(Y \wedge S) \vee (X \wedge \neg S)$
F	F	F	F	T	F	F
F	T	F	F	T	F	F
T	F	F	F	T	T	T
T	T	F	F	T	T	T
F	F	T	F	F	F	F
F	T	T	T	F	F	T
T	F	T	F	F	F	F
T	T	T	T	F	F	T



Two-input multiplexer

Thanks a lot



If you are taking a Nap, **wake up**.......Lecture Over