

CSE 390B, Spring 2023

Building Academic Success Through Bottom-Up Computing

Time Management & Boolean Arithmetic

Time Management, Overview of Numbers in Binary, Boolean Arithmetic, Circuits for Adding Binary Numbers

Project 2 Check-in

- ❖ How has Project 2 been coming along?
- ❖ What questions do you have about Project 2?
- ❖ Following the plan of action outlined in [slide 35](#) of last Thursday's lecture will work for implementing most chips
- ❖ Remember to double check your submission on GitLab
 - Navigate to GitLab, open tags, and verify that the associated commit includes your expected changes

Lecture Outline

❖ Time Management

- Identifying Weekly Time Commitments

❖ Overview of Numbers in Binary

- Comparison Between Binary and Decimal

❖ Boolean Arithmetic

- Addition Operator and Handling Binary Overflow

❖ Circuits for Adding Binary Numbers

- Overview of the Half Adder and Full Adder

Time Management

- ❖ One of your most valuable resources in college is time

- ❖ What typically fills up your time during the quarter?
 - Lectures, quiz sections, and attending office hours
 - Part-time jobs and working
 - Studying
 - Extracurricular activities or RSOs
 - Commuting
 - Chores at home
 - Socializing with friends and family
 - Physical, mental, spiritual activities

Weekly Time Commitments

- ❖ Class meeting times and quiz sections
- ❖ Family, friends, community, extracurricular commitments
- ❖ Physical, mental, social, spiritual activities
- ❖ Studying for each of your classes
 - The number of credits for a course reflects the number of hours the class meets
 - In general, courses require two hours of homework for every one hour of class
- ❖ What else is not reflected given your specific situation?

Tracking Weekly Time Commitments

- ❖ We often don't realize what takes up our time until we manually track how we use our time
- ❖ Exercise: Complete the weekly time commitments table
- ❖ Tip: Use different colors for different activity types

Weekly Time Commitments Table

Name: _____

Time	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
7:30am							
8:00am							
8:30am							
9:00am							
9:30am							
10:00am							
10:30am							
11:00am							
11:30am							
12:00pm							
12:30pm							
1:00pm							
1:30pm							
2:00pm							
2:30pm							
3:00pm							
3:30pm							
4:00pm							
4:30pm							
5:00pm							
5:30pm							
6:00pm							
6:30pm							
7:00pm							
7:30pm							
8:00pm							
8:30pm							
9:00pm							
9:30pm							
10:00pm							
10:30pm							
11:00pm							
11:30pm							

Time Management Group Discussion

Now that you've filled out your weekly time commitments, discuss the following in groups for 4-6 minutes:

- ❖ Does anything surprise you about the way you spend your time?
- ❖ What might you change about the way you utilize your time?
- ❖ How could you use this time commitments sheet in the future?

Lecture Outline

- ❖ Time Management
 - Identifying Weekly Time Commitments
- ❖ **Overview of Numbers in Binary**
 - **Comparison Between Binary and Decimal**
- ❖ Boolean Arithmetic
 - Addition Operator and Handling Binary Overflow
- ❖ Circuits for Adding Binary Numbers
 - Overview of the Half Adder and Full Adder

What is Binary?

- ❖ A **base n** number system is a system of number representation with n **symbols**
- ❖ Decimal system is a **base 10** number system
 - Base 10 symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 (each called a **digit**)
 - Increase a number by moving to the next greatest symbol
 - Add another digit when we run out of symbols
- ❖ Binary is a **base 2** number system
 - Base 2 symbols: 0, 1 (each called a **bit**)
 - Often prefixed with 0b (e.g., 0b1101, 0b10)
 - **Least-significant bit (LSB)**: Lowest-order position of a binary value
 - **Most-significant bit (MSB)**: Highest-order position of a binary value

Representing Numbers in Base 2

- ❖ Binary numbers are identical, except in base 2
 - Describe a value by specifying multiples of powers of 2
 - For example, a breakdown of 0b1101 in binary (13 in decimal)

Binary	Power of 2
0b1000	1×2^3
0b0100	1×2^2
0b0000	0×2^1
0b0001	1×2^0

Binary vs. Decimal

Binary	Decimal
0b000	0
0b001	1
0b010	2
0b011	3
0b100	4
0b101	5
0b110	6
0b111	7
...	...

< Lecture 3: Time Management & Boolean Arithmetic



🌐 When poll is active, respond at **PolleEv.com/cse390b**

What is the binary representation of the decimal value of 29?

0b011011

0b011101

0b100011

0b100111

We're lost...

Total Results: 0

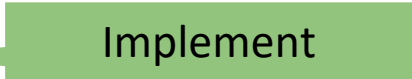

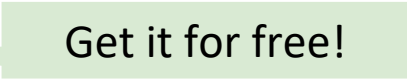
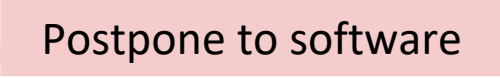
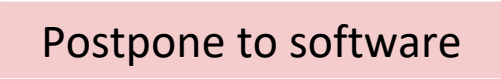
Powered by  **Poll Everywhere**



Lecture Outline

- ❖ Time Management
 - Identifying Weekly Time Commitments
- ❖ Overview of Numbers in Binary
 - Comparison Between Binary and Decimal
- ❖ **Boolean Arithmetic**
 - **Addition Operator and Handling Binary Overflow**
- ❖ Circuits for Adding Binary Numbers
 - Overview of the Half Adder and Full Adder

Roadmap: Boolean Arithmetic

- ❖ Addition  Implement
- ❖ Subtraction  Get it for free!
- ❖ Comparison ($<$, $>$, $==$, $!=$)  Get it for free!
- ❖ Multiplication  Postpone to software
- ❖ Division  Postpone to software

Binary Addition

- ❖ How do we add two binary numbers?
 - As humans, we could convert to decimal and then back
- ❖ Example: $0b101 + 0b010$
 - First convert $0b101$ to decimal (result is 5)
 - Next convert $0b010$ to decimal (result is 2)
 - Add the decimal numbers and convert back to binary
 - $5 + 2 = 7$, which is $0b111$ in binary
- ❖ What's more useful is understanding the rules of binary addition so we can teach them to a computer

Case Study: Decimal Addition

- ❖ Consider how we perform decimal addition
 - Right to left (least significant place to most significant place)
 - When a column's result is more than one digit, carry over the digit that overflows

❖ Example:

carry				
a	5	7	8	3
b	2	4	5	6
sum				

Binary Addition

- ❖ Binary addition conceptually the same as decimal addition
 - Right to left (least significant place to most significant place)
 - When a column's result is more than one digit, carry over the bit that overflows

- ❖ Example:

carry				
a	0	1	1	1
b	0	1	0	1
sum				

Binary Overflow

❖ What if there's a carry bit in the last column?

❖ Example:

carry				
a	0	1	1	0
b	1	0	1	0
sum				

Binary Overflow

- ❖ What if there's a carry bit in the last column?
- ❖ We can't represent it in our fixed-width numbers
 - We are going to “drop” or ignore the extra carry bit

carry					

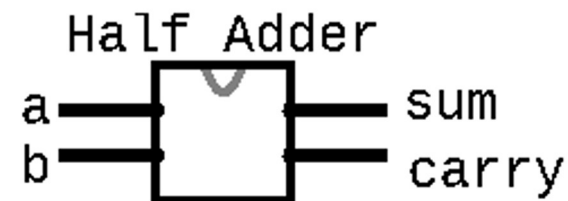
a		0	1	1	0
b		1	0	1	0
sum					

Lecture Outline

- ❖ Time Management
 - Identifying Weekly Time Commitments
- ❖ Overview of Numbers in Binary
 - Comparison Between Binary and Decimal
- ❖ Boolean Arithmetic
 - Addition Operator and Handling Binary Overflow
- ❖ **Circuits for Adding Binary Numbers**
 - **Overview of the Half Adder and Full Adder**

Half Adder

- ❖ Circuit for adding two bits together
- ❖ Takes in two inputs: `a`, `b`
 - `a` is the first bit being added
 - `b` is the corresponding bit to be added
- ❖ Produces two outputs: `sum`, `carry`
 - `sum` is the value to be put for this column in the result
 - `carry` is the value to be carried over to the next column



carry				
a	0	1	1	0
b	1	0	1	0
sum				

```
/**  
 * Computes the sum of 2 bits  
 */  
  
CHIP HalfAdder {  
  IN a, b;  
  OUT sum, carry;  
  
  PARTS:  
  // Put your code here:  
  
}
```

Half Adder Example

❖ Example: $0b0111 + 0b0101$

❖ For the right-most (least significant) column:

- $a = 1$
- $b = 1$
- $\text{sum} =$
- $\text{carry} =$

carry				
a	0	1	1	1
b	0	1	0	1
sum				

Half Adder Example

❖ Boolean expressions:

- $\text{sum} =$
- $\text{carry} =$

a	b	sum	carry
0	0		
0	1		
1	0		
1	1		

Half Adder Group Work

- ❖ Determine the half adder logical Boolean expression
 - First, fill in the truth table values for sum and carry based on the inputs
 - Then, develop a Boolean expression for sum and carry based on the truth table
- ❖ Five-minute group discussion, identify one person to share each of the following as a large group:
 - Overview of what the half adder does
 - Thought process for reaching the Boolean expression for sum
 - Thought process for reaching the Boolean expression for carry

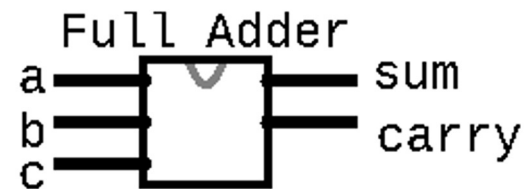
Half Adder Example

❖ Boolean expressions:

- $\text{sum} = a \text{ XOR } b$
- $\text{carry} = a \text{ AND } b$

a	b	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder



❖ Circuit for adding three bits together (two bits *and* carry bit together from previous column)

- `a` is the first bit being added
- `b` is the corresponding bit to be added
- `c` is the carry bit from the right column

carry				
a	0	1	1	0
b	1	0	1	0
sum				

❖ Produces two outputs: `sum`,
`carry`

- `sum` is the value to be put for this column in the result
- `carry` is the value to be carried over to the next column

```
/**
 * Computes the sum of 3 bits
 */

CHIP FullAdder {
  IN a, b, c;
  OUT sum, carry;

  PARTS:
  // Put your code here:

}
```

Full Adder

❖ Example: $0b0111 + 0b0101$

❖ For the second (second least significant) column:

- $a = 1$
- $b = 0$
- $c = 1$

▪ $\text{sum} =$

▪ $\text{carry} =$

carry			1	
<hr style="border-top: 1px dashed black;"/>				
a	0	1	1	1
b	0	1	0	1
sum				0

Full Adder Truth Table

a	b	c	sum	carry
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

< Lecture 3: Time Management & Boolean Arithmetic



Loading...

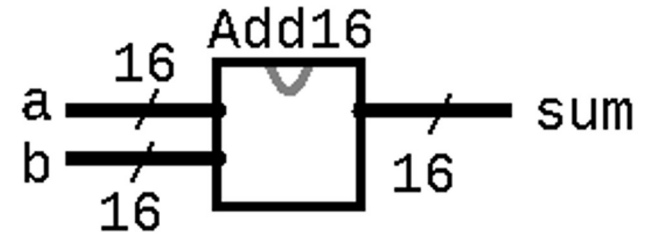


Full Adder Truth Table

a	b	c	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Multi-Bit Adder

- ❖ Adds two 16-bit numbers
- ❖ Connects the full adders for each column together (wires the out carry from one column to the in carry of the next)



```

/**
 * Adds two 16-bit Two's Complement
 * values. Overflow is ignored.
 */
CHIP Add16 {
    IN a[16], b[16];
    OUT sum[16];

    PARTS:
    // Put your code here:

}

```

...	0	1	1	1	0	0	
...	0	0	1	0	1	0	1
+	...	1	0	1	1	0	0
...	1	1	1	0	0	0	1

Post-Lecture 3 Reminders

- ❖ **Project 2 due Thursday (4/6) at 11:59pm**
- ❖ **Course Staff Support**
 - Preston has office hours in CSE2 153 today after lecture
 - Feel free to post your questions on the Ed discussion board too