Assignment 2: Converting Numbers & Truth Tables

Sarah Gillard

CS306: Computer Architecture

April 2, 2024

Part 1: Number Systems

You must show the logic you used to solve the following four problems:

Problem 1: Complete the following addition problem in hexadecimal: 32154AAAA + FEDCBA092. Show the answer in hexadecimal and in decimal.

We know that hexadecimal is also known as base 16. A, B, C, D, E, and F are used to represent values over nine in the hexadecimal system (Goodman, S., 2019). In hexadecimal, we have 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, where A is essentially equal to 10, B to 11, C to 12, D to 13, E to 14, and F to 15. As you can see, we don't go up to a letter equivalent to 16 because we start at 0 so there are 16 digits for us to use in hexadecimal values when we use 0-9 and A-F(equivalents of 10-15).

I set up a table to add the two numbers and keep track as we add each digit from right to left.

First, we will add A + 2. We know that A is equivalent to 10 here. 10 + 2 = 12. In hexadecimal, the letter C is equal to 12.

	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
									C

							1		
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
								3	C

Next, we will add A + 9. Again, we know that A is equivalent to 10. So 10 + 9 = 19. Hexadecimal goes up to F which has a value of 15. So here the 16 would take us back to 0, 17 back to 1, 18 back to 2, and 19 would get us back to 3. But we will regroup a 1 to the next column we add after placing the 3 as the result for this column.

Next, we will add the 1 we carried over with A + 0. So we have 10 + 1 = 11. B is the hexadecimal value for the number 11. So we simply put B as the result for this column's addition

							1		
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
							B	3	C

Next, we will add A + A, or 10 + 10. This, of course, equals 20. Since 20 is not a value in hexadecimal we need to find its equivalent in this base 16 number system. Hexadecimal goes up to F, or 15. So we are back to 0, 1, 2, 3, 4 to add 5 digits since 15 + 5 = 20. So, we will put a 4 in this column and regroup a 1 to the next column.

					1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
						4	B	3	C

Now, we will add 1 + 4 + B. B is equivalent to 11 in hexadecimal, so we add 1 + 4 + 11 which results in 16. So, we go back to 0 since hexadecimal stops at F which is equivalent to 15 and starts at 0. We need to regroup a 1 to the next column.

				1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
					0	4	B	3	C

Next, we need to add 1 + 5 + C. C is equivalent to 12 in hexadecimal so we have 1 + 5 + 12, which results in 18. So, we can get up to F or 15, but need to count 3 more to get to 18, starting back at 0. So we have 0, 1, 2. The result for this column is 2 and we need to regroup a 1 to the next column.

			1	1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
				2	0	4	B	3	C

Next, we add 1 + 1 + D. D is equivalent to 13, we have 2 + 13, which results in 15. 15 is equivalent to F in hexadecimal, we put F down as the sum for this column and there is no need to regroup as we don't need to start back at 0.

			1	1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
			F	2	0	4	B	3	C

Next, we add 2 + E, or 2 + 14, which sums to 16. Since F is equivalent to 15 we go back to the start of the hexadecimal system which is 0. We put 0 as the sum for this column and regroup a 1 to the next column.

	1		1	1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
		0	F	2	0	4	B	3	C

Finally, we add 1 + 3 + F. As we know, F is equivalent numerically to 15, we add 1 + 3 + 15. This results in 19. So we stop at 15 and count 4 more digits in the hexadecimal system starting back at 0. So we have 0, 1, 2, 3. Therefore this column results in a 3 and we must regroup a 1 into the next column.

1	1		1	1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
	3	0	F	2	0	4	B	3	C

As you can see, the result of adding the hexadecimal values 32154AAAA + FEDCBA092 equals 130F204B3C.

1	1		1	1	1	1			
	3	2	1	5	4	A	A	A	A
+	F	E	D	C	B	A	0	9	2
1	**3**	**0**	**F**	**2**	**0**	**4**	**B**	**3**	**C**

Now that we have the result in hexadecimal we need to show it in decimal. So now we will convert the hexadecimal value of 130F204B3C to decimal.

The first row in the table below represents each digit of our hexadecimal value 130F204B3C.

The second row represents the place value of the digit in base 16.

The third row shows how we must multiply the digit in that column by the place value of that digit.

Using a Ti-84 Plus calculator, we can solve the equations from the third row. This is the value in the fourth row.

1	3	0	F(15)	2	0	4	B(11)	3	C(12)
16^9	16^8	16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0
1x16^9	3x16^8	0x16^7	15x16^6	2x16^5	0x16^4	4x16^3	11x16^2	3x16^1	12x16^0
68,791,476,736	**12,884,901,888**	**0**	**251,658,240**	**2,097,152**	**0**	**16,384**	**2,816**	**48**	**12**

Now we add the results of the fourth column (in yellow), which results in 81,858,153,276. So, the hexadecimal 130F204B3C is equal to decimal 81,858,153,276.

Therefore, the answer to part 1, which instructs us to add these hexadecimal values and show the answer in hexadecimal and in decimal is that 32154AAAA + FEDCBA092 equals 130F204B3C in hexadecimal and 81,858,153,276 in decimal.

<u>Problem 2:</u> Convert the decimal number 4048891811 to hexadecimal.

Since hexadecimal is base 16, we will divide the decimal number 4,048,891,811 until the quotient becomes 0. I used my TI-84 Plus calculator for this.

4,048,891,811	/ 16	=	253,055,738	Remainder **3**
253,055,738	/ 16	=	15,815,983	Remainder 10 (**A** in hexadecimal)
15,815,983	/ 16	=	988,498	Remainder 15 (**F** in hexadecimal)
988,498	/ 16	=	61,781	Remainder **2**
3,861	/ 16	=	3,861	Remainder **5**
241	/ 16	=	241	Remainder **5**
241	/ 16	=	15	Remainder **1**
15	/ 16	=	0	Remainder 15 (**F** in hexadecimal)

Now, we need to add the reminders from top to bottom with the top number being the rightmost digit. I placed the top number in the right-most cell in the table below and added each digit going down the remainder and moving from right to left in the table. I added colors in the remainder list to match the table cell that the digit is in.

F	**1**	**5**	**5**	**2**	**F**	**A**	**3**

As you can see, the decimal value 4,048,891,811 is equivalent to the hexadecimal value F1552FA3.

<u>Problem 3:</u> Convert the octal number 2114112 to decimal.

We know that octal is also known as base 8. In the octal system, the value of each place is based on the powers of 8. To convert from octal to decimal, we use a process that involves both multiplication and division (Goodman, S., 2019).

The first row in the table below represents each digit of our octal value 2114112.

The second row represents the place value of the digit in base 8.

The third row shows how we must multiply the digit in that column by the place value of that digit.

Using a Ti-84 Plus calculator, we can solve the equations from the third row. This is the value in the fourth row.

2	1	1	4	1	1	2
8^6	8^5	8^4	8^3	8^2	8^1	8^0
2x8^6	1x8^5	1x8^4	4x8^3	1x8^2	1x8^1	2x8^0
524,288	32,768	4,096	2,048	64	8	2

Now we add the results of the fourth column:

		1	1	3	4	
						2
						8
					6	4
			2	0	4	8
			4	0	9	6
		3	2	7	6	8
+	5	2	4	2	8	8
	5	**6**	**3**	**2**	**7**	**4**

So, the octal value 2114112 is equal to the decimal value 563,274.

<u>Problem 4:</u> Expand the following table to include decimal numbers from 11 to 16, and expand the table to include hexadecimal numbers.

The table provided contained decimal numbers 0-16, binary 000-1010, and octal values 0-12. The additions I made to the table are in red.

Binary	Octal	Decimal	Hexadecimal	Notes To Show Work
000	0	0	0	
001	1	1	1	
010	2	2	2	
011	3	3	3	
100	4	4	4	
101	5	5	5	
110	6	6	6	
111	7	7	7	
1000	10	8	8	
1001	11	9	9	
1010	12	10	A	
1011	13	11	B	To convert 11 decimal to binary, divide 11 by 2 until the quotient becomes 0. 11/2 = 5 remainder 1. 5/2 = 2 remainder 1. 2/2 = 1 remainder 0. 2/1 = 0 remainder 1. The remainders from bottom to top are 1011. To convert 11 decimal to octal, divide 11 by 8 until the quotient becomes 0. 11/8 = 1R3. 1 / 8 = 0R1. From top to bottom, the remainders are 1 and 3, so the octal value is 13.

1100	**14**	12	**C**	To convert 12 decimal to binary, divide 12 by 2 until the quotient becomes 0. 12/2 = 6R0. 6/2 = 3R0. 3/2 = 1R1. 1 / 2 = 0R1. The remainders (R) from bottom to top are 1100. To convert 12 decimal to octal, divide 12 by 8 until the quotient becomes 0. 12/8 = 1R4. 1 / 8 = 0R1. From top to bottom, the remainders are 1 and 4, so the octal value is 14.
1101	**15**	13	**D**	To convert 13 decimal to binary, divide 13/2 = 6R1. 6/2 = 3R0. 3/2 = 1R1. 1 / 2 = 0R1. From bottom to top, the remainders are 1101. To convert 13 decimal to octal, divide 13 by 8 until the quotient becomes 0. 13/8 = 1R5. 1 / 8 = 0R1. From top to bottom, the remainders are 1 and 5, so the octal value is 15.
1110	**16**	14	**E**	To convert 14 decimal to binary, divide 14/2 = 7R0. 7/2 = 3R1. 3/2 = 1R1. 1 / 2 = 0R1. From top to bottom, the remainders are 1110. To convert 14 decimal to octal, divide 14 by 8 until the quotient becomes 0. 14/8 = 1R6. 1 / 8 = 0R1. From top to bottom, the remainders are 1 and 6, so the octal value is 16.
1111	**17**	15	**F**	To convert 15 decimal to binary, divide 15/2 = 7R1. 7/2 = 3R1. 3/2 = 1R1. 1 / 2 = 0R1. From top to bottom, the remainders are 1111. To convert 15 decimal to octal, divide 15 by 8 until the quotient becomes 0. 15/8 = 1R7. 1 / 8 = 0R1. From top to bottom, the remainders are 1 and 7, so the octal value is 17.
10000	**20**	16	**10**	To convert 16 decimal to binary, divide 16/2 = 8R0. 8/2 = 4R0. 4/2 = 2R0. 2/2 = 1R0. 2/1 = 0R1. From top to bottom, the reminders are 10000. Hexadecimal is base 16. Divide 16 by 16. The result is 1 remainder 0. Divide 16 by 1 and the result is 0 remainder 1. From the bottom to the top, the reminders are 1 and 0, or 10.

				To convert 16 decimal to octal, divide 16 by 8 until the quotient becomes 0. 16/8 = 2R0. 2/8 = 0R2. From the bottom to the top, the remainders are 2 and 0. So the octal value is 20.

Part 2: Truth Tables and Logic Gates

State the Boolean Expression and truth table for the following logic gates:

NAND

NOR

XOR

NOT

3-input AND gate (inputs A, B, C, and output X)

NAND truth table

NAND Boolean Expression: Z = $\overline{X}$ **+** $\overline{Y}$

X	Y	$\overline{X.Y}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR truth table

NOR Boolean Expression: Z = $\overline{X + Y}$

X	Y	$\overline{X + Y}$
0	0	1
0	1	0
1	0	0
1	1	0

XOR truth table

XOR Boolean Expression: $Z = X \oplus Y = \overline{X} \bullet Y + X \bullet \overline{Y}$

X	Y	$X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

NOT truth table

NOT Boolean Expression: $X = \overline{X}$

X	$\overline{X}$
0	1
1	0

3-input AND gate

Boolean Expression: $X = A \bullet B \bullet C$

(inputs A, B, C, and output X) truth table

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
1	0	0	0
0	1	1	0
1	1	0	0
1	0	1	0
1	1	1	1

Part 3: Reflection

Write a reflection consisting of 500-750 words that reflects upon this question: Why is it important to study how to manipulate fixed-point numbers? Provide examples of these systems in use and use scholarly research in your answer.

Understanding the manipulation of fixed-point numbers is crucial across different fields as these values help in maintaining precision in various computations. Fixed-point numbers are a subset of rational numbers, having a limited number of digits before and after a decimal point (Fox, C., 2024). These numbers find their way into many different areas, from finance to computer graphics and digital signal processing (Anon., 2015).

Fixed-point values are used in the world of finance when working with currency, interest rates, and stock prices. These calculations all rely heavily on fixed-point arithmetic to make sure everything's accurate. In the U.S., for example, currency values always have to be precise up to two decimal places. And when it comes to interest rates and stock prices, any miscalculations could lead to big problems. So, knowing how to handle fixed-point numbers properly is a must.

In computer graphics, fixed-point numbers are important for things like representing coordinates and determining where objects go on the screen. Whether it's a detailed 3D scene or a simple 2D animation, getting the coordinates right is essential for making everything look good. Fixed-point arithmetic helps ensure graphics come out looking just the way they should (Kim, G., 2024).

Fixed-point values are also important in digital audio and image processing. When working with sound or images, precision is essential. Fixed-point arithmetic helps keep things accurate during these calculations, such as calculations to filter out noise or enhance image quality (Kim, G., 2024). It helps ensure everything sounds and looks just right.

Knowing how to manipulate fixed-point numbers is also essential when converting between binary and decimal numbers. This is something you could encounter while working in digital systems. Understanding fixed-point arithmetic is key. Take the example of converting a fixed-point binary number to a decimal value. Each digit's position relative to the decimal point tells you how much it contributes to the overall value. By following a systematic process of multiplication and addition based on the digit's position, you can get the decimal representation spot on (Fox, C., 2024).

To convert a fixed point binary number to its decimal value, all digits to the left of the decimal should be multiplied times 2^n where n = 0, 1, 2, 3, etc. increasing from right to left (Fogle, A., 2021). All digits to the right of the decimal should be multiplied by 2^-n where n = 0, 1, 2, 3, etc. increasing from left to right. For example, in the binary fixed point number 01100.0110 the decimal equivalent can be found by using this information to convert the numbers to the left of the decimal and then the numbers to the right of the decimal.

To the left of the decimal is the value 011000. The first row in the table below represents each digit of the binary value to the left of the decimal. The second row represents the place value of the digit in base 2. The third row shows how to multiply the digit in that column by the place value of that digit. Lastly, add the values in the fourth row together, which results in 56.

0	1	1	0	0	0
2^5	2^4	2^3	2^2	2^1	2^0
1x2^5	1x2^4	1x2^3	0x2^2	0x2^1	0x2^0
32	16	8	0	0	0

So, the decimal value of the binary number to the right of the decimal is 56. Now, convert the binary number to the right of the decimal. To the right of the decimal, are the binary numbers- 0110.

0	1	1	0
2^-3	2^-2	2^-1	2^0
0x2^-3	1x2^-2	1x2^-1	0x2^0
0	.25	.5	0

Add the values together, .25 + .5. This results in .75. So, .75 is the decimal value after converting the binary value to the right of the decimal. Putting these values together, 56.75 is the decimal equivalent of the binary number 011000.0110.

Fixed point values can also be converted from decimal to binary. For example, convert 7.375 to binary by first converting the 7 to binary using the usual decimal-to-binary conversion. For the numbers to the right of the decimal, multiply by 2. To convert 7 to binary, solve 7/2 which is 3 remainder 1. Now solve 3/2 which is 1 remainder 1. Next, solve 1 divided by 2, which is 0 remainder 1. Now, read the remainders from bottom to top: 111. 7 in binary is 111.

Now, look at the value to the right of the decimal, .375. To convert this number to binary, multiply the decimal value by 2. .375x2 = 0.75. There is no whole number so add a 0 to the right of the decimal point. Next, .75x2 = 1.5. Place the 1 to the right of the binary fixed point values calculated so far .01. Next, multiply the value to the right of the decimal point from 1.5, so .5x2 = 1. This is a whole number and it is added to the right of the binary values calculated so far, resulting in .011. So the binary value to the right of the decimal point is .011. So, the decimal value 7.375 is equivalent to the binary value 111.011.

Mastering fixed-point manipulation is essential across a wide range of disciplines. Whether you're dealing with finance, graphics, signal processing, or numerical conversions, having a solid understanding of fixed-point arithmetic is key to ensuring accuracy and precision in your work. Fixed-point arithmetic helps you represent the fractional parts accurately, ensuring that your conversions are precise (Fox, C., 2024). It helps drive precision in numerical computing across the board.

References

Anonymous. *Boolean algebra.* Practical EE. https://practicalee.com/boolean-algebra/

Anonymous. (2015, December 2). *Fixed-point vs. floating-point digital signal processing.* Analog Devices. https://www.analog.com/en/resources/technical-articles/fixedpoint-vs-floatingpoint-dsp.html

Arizzi, R. (2018, May 13). *Electric gate symbols for boolean functions.* Study.com. https://study.com/academy/lesson/electric-gate-symbols-for-boolean-functions.html

Fogle, A. (2021, November 12). *Converting fixed point values in the binary numerical system*. Study.com. https://study.com/academy/lesson/converting-fixed-point-values-in-the-binary-numerical-system.html

Fox, Charles. (2024). *Computer Architecture: From The Stone Age to the Quantum Age.* No Starch Press.

Goodman, S. (2019, August 31). *Converting between binary, decimal, octal & hexadecimal numbers*. Study.com. https://study.com/academy/lesson/converting-between-binary-decimal-octal-hexadecimal-numbers.html

Kim, Geoffrey. (2024, March 29). *Decoding numerical representation: floating-point vs. fixed-point arithmetic in computing.* Dev.to. https://dev.to/mochafreddo/decoding-numerical-representation-floating-point-vs-fixed-point-arithmetic-in-computing-3h46

Yang Alcocer, Y. (2013, November 7). *Truth value | definition, propositions & tables.* Study.com. https://study.com/learn/lesson/propositions-truth-value-tables.html