CO450 Computer Architectures Exercise Workbook

Compiled by Justin Luker

Class Exercises for Week 2

Re-Cap on Base 10 Decimal Numbers

1. What is the correct way to write out the following base 10 number:

45_{10}

				(4×10^1)	+	(5×10^0)	=	45_{10}

2. What is the correct way to write out the following base 10 number:

68_{10}

3. What is the correct way to write out the following base 10 number:

183_{10}

4. What is the correct way to write out the following base 10 number:

3549_{10}

5. What is the correct way to write out the following base 10 number:

27318_{10}

Positional Notation: Binary to Decimal Conversion

1. Convert the following binary number to decimal using the Positional Notation method:

00001110_2

We have worked this first question through for you.

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	**0**	**0**	**0**	**1**	**1**	**1**	**0**
8 + 4 + 2 = 14							

The correct answer is:

14_{10}

2. Convert the following binary number to decimal using the Positional Notation method:

00101010_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

3. Convert the following binary number to decimal using the Positional Notation method:

10001101_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

4. Convert the following binary number to decimal using the Positional Notation method:

11011111_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

5. Convert the following binary number to decimal using the Positional Notation method:

01110001_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

6. Convert the following binary number to decimal using the Positional Notation method:

11101010_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Doubling: Binary to Decimal Conversion

1. Convert the following binary number to decimal using the Doubling method:

$\mathbf{10110100_2}$

We have worked this first question through for you.

0	x	2	+	1	=	1
1	x	2	+	0	=	2
2	x	2	+	1	=	5
5	x	2	+	1	=	11
11	x	2	+	0	=	22
22	x	2	+	1	=	45
45	x	2	+	0	=	90
90	x	2	+	0	=	**180**

The answer is:

$\mathbf{180_{10}}$

2. Convert the following binary number to decimal using the Doubling method:

$\mathbf{11100111_2}$

0	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	

The answer is:

3. Convert the following binary number to decimal using the Doubling method:

00110101_2

0	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	

The answer is:

4. Convert the following binary number to decimal using the Doubling method:

00111000_2

0	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	

The answer is:

Short Division by Two with Remainder: Decimal to Binary Conversion

1. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

86_{10}

We have worked this first question through for you.

86	/	2	=	43	Remainder	0
43	/	2	=	21	Remainder	1
21	/	2	=	10	Remainder	1
10	/	2	=	5	Remainder	0
5	/	2	=	2	Remainder	1
2	/	2	=	1	Remainder	0
1	/	2	=	0	Remainder	1
	/	2	=		Remainder	

The answer is:

1010110_2

2. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

109_{10}

	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	

The answer is:

3. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

72_{10}

	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	

The answer is:

4. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

124_{10}

	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	

The answer is:

Comparison with Descending Powers of Two and Subtraction: Decimal to Binary Conversion

1. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

57_{10}

We have worked this first question through for you.

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	**0**	**1**	**1**	**1**	**0**	**0**	**1**
		57 - 32 = 25	**25 - 16 = 9**	**9 - 8 = 1**			**1-1 = 0**

The answer is:

0	0	1	1	1	0	0	1

2. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

113_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The answer is:

3. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

93_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The answer is:

4. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

29_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The answer is:

Binary Addition

1. Add the following binary numbers together, what is the correct answer:

$00000110_2 + 00001010_2 =$

We have worked this first question through for you.

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	0	**0**	**0**	**0**	**0**	**1**	**1**	**0**
+	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**0**
	0	**0**	**0**	**1**	**0**	**0**	**0**	**0**
				1	1	1		

The answer is:

0	**0**	**0**	**1**	**0**	**0**	**0**	**0**

2. Add the following binary numbers together, what is the correct answer:

$00000110_2 + 00011100_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The answer is:

3. Add the following binary numbers together, what is the correct answer:

$00101100_2 + 00111001_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The answer is:

4. Add the following binary numbers together, what is the correct answer:

$00111100_2 + 00010101_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The answer is:

The Answers for Week 2

Re-Cap on Base 10 Decimal Numbers

1. $(4 \times 10^1) + (5 \times 10^0) = 45_{10}$

2. $(6 \times 10^1) + (8 \times 10^0) = 68_{10}$

3. $(1 \times 10^2) + (8 \times 10^1) + (3 \times 10^0) = 183_{10}$

4. $(3 \times 10^3) + (5 \times 10^2) + (4 \times 10^1) + (9 \times 10^0) = 3549_{10}$

5. $(2 \times 10^4) + (7 \times 10^3) + (3 \times 10^2) + (1 \times 10^1) + (8 \times 10^0) = 27318_{10}$

Positional Notation: Binary to Decimal Conversion

1. 14_{10}

2. 42_{10}

3. 141_{10}

4. 223_{10}

5. 113_{10}

6. 234_{10}

Doubling: Binary to Decimal Conversion

1. 180_{10}

2. 231_{10}

3. 53_{10}

4. 56_{10}

Short Division by Two with Remainder: Decimal to Binary Conversion

1. 01010110_2

2. 01101101_2

3. 01001000_2

4. 01111100_2

Comparison with Descending Powers of Two and Subtraction

1. 00111001_2

2. 01110001_2

3. 01011101_2

4. 00011101_2

Binary Addition

1. 00010000_2

2. 00100010_2

3. 01100101_2

4. 01010001_2

Class Exercises for Week 3

Recap on Binary to Decimal Conversion

1. Convert the following binary number to decimal using the Positional Notation method:

11101010_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

2. Convert the following binary number to decimal using the Doubling method:

00111101_2

0	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	

The correct answer is:

Recap on Decimal to Binary Conversion

1. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

167_{10}

	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	

The correct answer is:

2. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

147_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Binary Addition

1. Add the following binary numbers together, what is the correct answer:

$00110110_2 + 00101110_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The correct answer is:

2. Add the following binary numbers together, what is the correct answer:

$10100111_2 + 00010111_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The correct answer is:

Binary Multiplication

1. Multiply the following binary numbers, what is the correct answer:

00010110_2 X 00000110_2 =

We have worked this first question through for you.

				128	64	32	16	8	4	2	1
				2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
X				**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**
				0	**0**	**0**	**0**	**0**	**1**	**1**	**0**
				0	**0**	**0**	**0**	**0**	**0**	**0**	**0**
+			**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**0**
		0	**0**	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**
		0	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
				1	1	1	1				

The correct answer is:

1	**0**	**0**	**0**	**0**	**1**	**0**	**0**

2. Multiply the following binary numbers, what is the correct answer:

00101100_2 X 00000101_2 =

				128	64	32	16	8	4	2	1
				2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
X											
+											

The answer is:

3. Multiply the following binary numbers, what is the correct answer:

00000110_2 X 00001101_2 =

				128	64	32	16	8	4	2	1
				2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
X											
+											

The correct answer is:

4. Multiply the following binary numbers, what is the correct answer:

00000101_2 X 00001010_2 =

				128	64	32	16	8	4	2	1
				2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
X											
+											

The correct answer is:

Two's Complement

1. Convert 35_{10} to binary then use Two's Complement to convert the binary representation of 35_{10} in to the Two's Complemented binary representation for -35_{10}, what is the correct answer:

We have worked this first question through for you.

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**1**
Flipped bits	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**
One to add to the flipped bits above	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Result of addition of flipped bits and one	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**
Carry Bits								

The correct answer is:

1	**1**	**0**	**1**	**1**	**1**	**0**	**1**

2. Convert 106_{10} to binary then use Two's Complement to convert the binary representation of 106_{10} in to the Two's Complemented binary representation for -106_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

3. Convert 73_{10} to binary then use Two's Complement to convert the binary representation of 73_{10} in to the Two's Complemented binary representation for -73_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

4. Convert 93_{10} to binary then use Two's Complement to convert the binary representation of 93_{10} in to the Two's Complemented binary representation for -93_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Binary Subtraction

1. Subtract 35_{10} from 100_{10} in binary, what is the correct answer:

$100_{10} - 35_{10} =$

We have worked this first question through for you.

		128	64	32	16	8	4	2	1
		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert **First Term** decimal to binary		**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
Positional notation used to convert **Second Term** decimal to binary		**0**	**0**	**1**	**0**	**0**	**0**	**1**	**1**
Flipped bits of **Second Term**		**1**	**1**	**0**	**1**	**1**	**1**	**0**	**0**
One to add to the flipped bits of **Second Term**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**
Two's Complement of **Second Term**		**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**
Carry Bits									
Addition of **First Term** and Two's Complement of **Second Term**	**+**	**0**	**1**	**1**	**0**	**0**	**1**	**0**	**0**
		1	**1**	**0**	**1**	**1**	**1**	**0**	**1**
Result	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**
Carry Bits	**1**	**1**	**1**	**1**	**1**	**1**			

The correct answer is:

0	**1**	**0**	**0**	**0**	**0**	**0**	**1**

2. Subtract 28_{10} from 87_{10} in binary, what is the correct answer:

$87_{10} - 28_{10} =$

		128	64	32	16	8	4	2	1
		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert **First Term** decimal to binary									
Positional notation used to convert **Second Term** decimal to binary									
Flipped bits of **Second Term**									
One to add to the flipped bits of **Second Term**									
Two's Complement of **Second Term**									
Carry Bits									
Addition of **First Term** and Two's Complement of **Second Term**	+								
Result									
Carry Bits									

The correct answer is:

3. Subtract 49_{10} from 77_{10} in binary, what is the correct answer:

$77_{10} - 49_{10} =$

		128	64	32	16	8	4	2	1
		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert **First Term** decimal to binary									
Positional notation used to convert **Second Term** decimal to binary									
Flipped bits of **Second Term**									
One to add to the flipped bits of **Second Term**									
Two's Complement of **Second Term**									
Carry Bits									
Addition of **First Term** and Two's Complement of **Second Term**	+								
Result									
Carry Bits									

The correct answer is:

4. Subtract 63_{10} from 115_{10} in binary, what is the correct answer:

115_{10} - 63_{10} =

		128	64	32	16	8	4	2	1
		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert **First Term** decimal to binary									
Positional notation used to convert **Second Term** decimal to binary									
Flipped bits of **Second Term**									
One to add to the flipped bits of **Second Term**									
Two's Complement of **Second Term**									
Carry Bits									
Addition of **First Term** and Two's Complement of **Second Term**	+								
Result									
Carry Bits									

The correct answer is:

Converting Hexadecimal to Decimal

1. Convert the following Hexadecimal (base 16) number to Decimal (base 10):

$A16_{16}$

We have worked this first question through for you.

+	$(10_{10} \times 256_{10})$		2	5	6	0
	$(1_{10} \times 16_{10})$				1	6
	$(6_{10} \times 1_{10})$					6
	=		2	5	8	2
					1	

The correct answer is:

2582_{10}

2. Convert the following Hexadecimal (base 16) number to Decimal (base 10):

$4DF_{16}$

+						
	=					

The correct answer is:

3. Convert the following Hexadecimal (base 16) number to Decimal (base 10):

$E5C_{16}$

+						
	=					

The correct answer is:

4. Convert the following Hexadecimal (base 16) number to Decimal (base 10):

135_{16}

+						
	=					

The correct answer is:

Converting Hexadecimal to Binary

1. Convert the following Hexadecimal (base 16) number to Binary (base 2):

$0A16_{16}$

We have worked this first question through for you.

	8	4	2	1
	2^3	2^2	2^1	2^0
6_{16}	**0**	**1**	**1**	**0**
1_{16}	**0**	**0**	**0**	**1**
A_{16}	**1**	**0**	**1**	**0**
0_{16}	**0**	**0**	**0**	**0**

The correct answer is:

0	**0**	**0**	**0**	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**0**

2. Convert the following Hexadecimal (base 16) number to Binary (base 2):

$FE37_{16}$

	8	4	2	1
	2^3	2^2	2^1	2^0

The correct answer is:

Converting Binary to Hexadecimal

1. Convert the following Binary (base 2) number to Hexadecimal (base 16):

1101000100100110_2

We have worked this first question through for you.

	8	4	2	1
	2^3	2^2	2^1	2^0
6_{16}	**0**	**1**	**1**	**0**
2_{16}	**0**	**0**	**1**	**0**
1_{16}	**0**	**0**	**0**	**1**
D_{16}	**1**	**1**	**0**	**1**

The correct answer is:

$D126_{16}$

2. Convert the following Binary (base 2) number to Hexadecimal (base 16):

1011011011100111_2

	8	4	2	1
	2^3	2^2	2^1	2^0

The correct answer is:

Converting Octal to Decimal

1. Convert the following Octal (base 8) number to Decimal (base 10):

217_8

We have worked this first question through for you.

+	(2_{10} x 64_{10})			1	2	8
	(1_{10} x 8_{10})					8
	(7_{10} x 1_{10})					7
	=			1	4	3
					2	

The correct answer is:

143_{10}

2. Convert the following Octal (base 8) number to Decimal (base 10):

435_8

+						
	=					

The correct answer is:

Converting Octal to Binary

1. Convert the following Octal (base 8) number to Binary (base 2):

761_8

We have worked this first question through for you.

	4	2	1
	2^2	2^1	2^0
1_8	**0**	**0**	**1**
6_8	**1**	**1**	**0**
7_8	**1**	**1**	**1**

The answer is:

0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**1**

2. Convert the following Octal (base 8) number to Binary (base 2):

357_8

	4	2	1
	2^2	2^1	2^0

The answer is:

Converting Binary to Octal

1. Convert the following Binary (base 2) number to Octal (base 8):

$0110\ 110\ 101\ 010\ 011_2$

We have worked this first question through for you.

	4	2	1
	2^2	2^1	2^0
3_8	**0**	**1**	**1**
2_8	**0**	**1**	**0**
5_8	**1**	**0**	**1**
6_8	**1**	**1**	**0**
6_8	**1**	**1**	**0**
0_8	**-**	**-**	**0**

The correct answer is:

66523_8

2. Convert the following Binary (base 2) number to Octal (base 8):

0111101101110100_2

	4	2	1
	2^2	2^1	2^0

The correct answer is:

The Answers for Week 3

Recap on Binary to Decimal Conversion

1. 234_{10}

2. 61_{10}

Recap on Decimal to Binary Conversion

1. 10100111_2

2. 10010011_2

Recap on Binary Addition

1. 01100100_2

2. 10111110_2

Binary Multiplication

1. 10000100_2

2. 11011100_2

3. 01001110_2

4. 00110010_2

Two's Complement

1. 11011101_2

2. 10010110_2

3. 10110111_2

4. 10100011_2

Binary Subtraction

1. 01000001_2

2. 00111011_2

3. 00011100_2

4. 00110100_2

Converting Hexadecimal to Decimal

1. 2582_{10}

2. 1247_{10}

3. 3676_{10}

4. 309_{10}

Converting Hexadecimal to Binary

1. 0000101000010110_2

2. 1111111000110111_2

Converting Binary to Hexadecimal

1. $D126_{16}$

2. $B6E7_{16}$

Converting Octal to Decimal

1. 143_{10}

2. 285_{10}

Converting Octal to Binary

1. 0000000111110001_2

2. 0000000011101111_2

Converting Binary to Octal

1. 66523_8

2. 75564_8

Class Exercises for Week 4

Recap on Binary to Decimal Conversion

1. Convert the following binary number to decimal using the Positional Notation method:

10001110_2

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

2. Convert the following binary number to decimal using the Doubling method:

11001100_2

0	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	
	x	2	+		=	

The correct answer is:

Recap on Decimal to Binary Conversion

1. Convert the following decimal number to binary using the Short Division by Two with Remainder method:

185_{10}

	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	
	/	2	=		Remainder	

The correct answer is:

2. Convert the following decimal number to binary using the Comparison with Descending Powers of Two and Subtraction method:

72_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Binary Addition

1. Add the following binary numbers together, what is the correct answer:

$00111100_2 + 00111010_2 =$

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+								

The correct answer is:

Recap on Binary Multiplication

1. Multiply the following binary numbers, what is the correct answer:

$00010110_2 \times 00000110_2 =$

				128	64	32	16	8	4	2	1
				2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
X											
+											

The correct answer is:

Recap on Two's Complement

1. Convert 112_{10} to binary then use Two's Complement to convert the binary representation of 112_{10} in to the Two's Complemented binary representation for -112_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

2. Convert 49_{10} to binary then use Two's Complement to convert the binary representation of 49_{10} in to the Two's Complemented binary representation for -49_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Recap on Binary Subtraction

1. Subtract 93_{10} from 187_{10} in binary, what is the correct answer:

187_{10} - 93_{10} =

		128	64	32	16	8	4	2	1
		2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert **First Term** decimal to binary									
Positional notation used to convert **Second Term** decimal to binary									
Flipped bits of **Second Term**									
One to add to the flipped bits of **Second Term**									
Two's Complement of **Second Term**									
Carry Bits									
Addition of **First Term** and Two's Complement of **Second Term**	**+**								
Result									
Carry Bits									

The correct answer is:

Recap on Converting Hexadecimal to Decimal

1. Convert the following Hexadecimal (base 16) number to Decimal (base 10):

$1EA_{16}$

+						
	=					

The correct answer is:

Recap on Converting Hexadecimal to Binary

1. Convert the following Hexadecimal (base 16) number to Binary (base 2):

$A4B9_{16}$

	8	4	2	1
	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Converting Binary to Hexadecimal

1. Convert the following Binary (base 2) number to Hexadecimal (base 16):

0010010011101111_2

	8	4	2	1
	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Converting Octal to Decimal

1. Convert the following Octal (base 8) number to Decimal (base 10):

372_8

+						
	=					

The correct answer is:

Recap on Converting Octal to Binary

1. Convert the following Octal (base 8) number to Binary (base 2):

245_8

[redacted]	4	2	1
	2^2	2^1	2^0

The correct answer is:

Recap on Converting Binary to Octal

1. Convert the following Binary (base 2) number to Octal (base 8):

0101011101110111_2

[redacted]	4	2	1
	2^2	2^1	2^0

The correct answer is:

Signed Magnitude Notation

1. Represent the following decimal number in binary using Signed Magnitude Notation:

-104_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

2. Represent the following decimal number in binary using Signed Magnitude Notation:

98_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

3. Represent the following decimal number in binary using Signed Magnitude Notation:

-76_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Binary Excess Notation to Decimal

1. What is the decimal number that is represented by 10100000_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	1	**0**	**1**	**0**	**0**	**0**	**0**	**0**
Unsigned binary to decimal conversion using positional notation	**$128 + 32 = 160_{10}$**							
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits	**$160 - 2^7 = 32_{10}$**							

The correct answer is:

32_{10}

2. What is the decimal number that is represented by 01101100_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

3. What is the decimal number that is represented by 11100111_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

Decimal to Binary Excess Notation

1. What is the binary Excess Notation representation of the following decimal number:

-20_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: *n* = number of bits	**$-20 + 128 = 108_{10}$**							
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**0**

The correct answer is:

0	**1**	**1**	**0**	**1**	**1**	**0**	**0**

2. What is the binary Excess Notation representation of the following decimal number:

-93_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: *n* = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

The Answers for Week 4

Recap on Binary to Decimal Conversion

1. 142_{10}

2. 204_{10}

Recap on Decimal to Binary Conversion

1. 10111001_{2}

2. 01001000_{2}

Recap on Binary Addition

1. 01110110_{2}

Recap on Binary Multiplication

1. 10000100_{2}

Recap on Two's Complement

1. 10010000_{2}

2. 11001111_{2}

Recap on Binary Subtraction

1. 01011110_{2}

Recap on Converting Hexadecimal to Decimal

1. 490_{10}

Recap on Converting Hexadecimal to Binary

1. 1010010010111001_{2}

Recap on Converting Binary to Hexadecimal

1. $24EF_{16}$

Recap on Converting Octal to Decimal

1. 250_{10}

Recap on Converting Octal to Binary

1. 0000000010100101_{2}

Recap on Converting Binary to Octal

1. 53567_{8}

Signed Magnitude Notation

1. 11101000_{2}

2. 01100010_2

3. 11001100_2

Binary Excess Notation to Decimal

1. 32_{10}

2. -20_{10}

3. 103_{10}

Decimal to Binary Excess Notation

1. 01101100_2

2. 00100011_2

Class Exercises for Week 5

Signed Magnitude Notation

1. Represent the following decimal number in binary using Signed Magnitude Notation:

-62_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

2. Represent the following decimal number in binary using Signed Magnitude Notation:

14_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

3. Represent the following decimal number in binary using Signed Magnitude Notation:

-103_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Binary Excess Notation to Decimal

1. What is the decimal number that is represented by 10100010_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	1	**0**	**1**	**0**	**0**	**0**	**1**	**0**
Unsigned binary to decimal conversion using positional notation	**$128 + 32 + 2 = 162_{10}$**							
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits	**$162 - 2^7 = 34_{10}$**							

The correct answer is:

34_{10}

2. What is the decimal number that is represented by 01101100_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

3. What is the decimal number that is represented by 11100111_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

Decimal to Binary Excess Notation

1. What is the binary Excess Notation representation of the following decimal number:

-50_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: *n* = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

2. What is the binary Excess Notation representation of the following decimal number:

-83_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: *n* = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

Two's Complement

1. Convert 26_{10} to binary then use Two's Complement to convert the unsigned binary representation of 26_{10} in to the Two's Complemented binary representation for -26_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

2. Convert 118_{10} to binary then use Two's Complement to convert the unsigned binary representation of 118_{10} in to the Two's Complemented binary representation for -118_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Two's Complement Binary Additions

1. Add the following numbers together using two's complement binary representation and then answer the questions below:

80_{10} + -33_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

2. Add the following numbers together using two's complement binary representation and then answer the questions below:

-104_{10} + -32_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

3. Add the following numbers together using two's complement binary representation and then answer the questions below:

73_{10} + 12_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

4. Add the following numbers together using two's complement binary representation and then answer the questions below:

-11_{10} + -28_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

Decimal to Excess 50 Notation

1. Represent 15_{10} in excess 50 notation:

We have done this first one for you.

15_{10}	+	50_{10}	=	65_{10}

The correct answer is:

65_{10}

2. Represent -38_{10} in excess 50 notation:

	+	50_{10}	=	

The correct answer is:

3. Represent 47_{10} in excess 50 notation:

	+	50_{10}	=	

The correct answer is:

4. Represent -50_{10} in excess 50 notation:

	+	50_{10}	=	

The correct answer is:

Conversion of Decimal Numbers to SEEZMMMM Format

1. Convert -12,658$_{10}$ into the SEEZMMMM format:

We have done this first one for you.

S = signed magnitude notation	=	1
EE = Exponent Excess 50 notation	=	5 + 50 = 55
ZMMMM = Mantissa	=	12658
Normalise Mantissa	=	12658

The correct answer is:

15512658

2. Convert +0.05432$_{10}$ into the SEEZMMMM format:

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

3. Convert -105.678$_{10}$ into the SEEZMMMM format:

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

4. Convert -1,542.768$_{10}$ into the SEEZMMMM format:

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

Conversion of SEEZMMMM Format to Decimal Number

1. Convert 14667840 in the SEEZMMMM Format to a decimal number:

We have done this first one for you.

S = signed magnitude notation	=	1	=	-
EE = Exponent Excess 50 notation	=	46	=	10^{-4}
ZMMMM = Mantissa	=	67840	=	$.6784 \times 10^{-4} = 0.000067840$

The correct answer is:

- 0.000067840

2. Convert 05617683 in the SEEZMMMM Format to a decimal number:

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

3. Convert 14275355 in the SEEZMMMM Format to a decimal number:

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

4. Convert 16144387 in the SEEZMMMM Format to a decimal number:

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

Conversion of Decimal Exponent to Excess 127 Binary

1. Represent an Exponent of -8_{10} in the Excess 127 format:

We have done this one for you.

Excess	127	=	0	1	1	1	1	1	1	1
Two's Complement Exponent	-8	=	1	1	1	1	1	0	0	0
Result of addition of excess and exponent			0	1	1	1	0	1	1	1
Carry's		*1*	*1*	*1*	*1*	*1*				

The correct answer is:

01110111_2	=	119_{10}

2. Represent an Exponent of -4_{10} in the Excess 127 format:

Excess	127	=								
Two's Complement Exponent		=								
Result of addition of excess and exponent										
Carry's										

The correct answer is:

	=	

3. Represent an Exponent of 12_{10} in the Excess 127 format:

Excess	127	=								
Two's Complement Exponent		=								
Result of addition of excess and exponent										
Carry's										

The correct answer is:

	=	

Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. Convert the following IEEE 754 single precision binary float to decimal:

$01000011111000000000000000000000_2$

We have done this one for you.

Sign	0	=	+							
Excess		=	1	0	0	0	0	1	1	1
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=	$135_{10} – 127_{10} = 8_{10}$							

Mantissa	$1.11000000000000000000000000000000_2$
Shift decimal point with exponent	$111000000.00000000000000000000000_2$
Convert shifted binary mantissa to decimal	$256_{10} + 128_{10} + 64_{10} = 448_{10}$
Add sign to converted decimal	$+448_{10}$

The correct answer is:

$+448_{10}$

2. Convert the following IEEE 754 single precision binary float to decimal:

$11000001111010000000000000000000_2$

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

3. Convert the following IEEE 754 single precision binary float to decimal:

$01000010110110000000000000000000_2$

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

4. Convert the following IEEE 754 single precision binary float to decimal:

$01000010010011000000000000000000_2$

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

The Answers for Week 5

Signed Magnitude Notation

1. 10111110

2. 00001110

3. 11100111

Binary Excess Notation to Decimal

1. 34

2. -20

3. 103

Decimal to Binary Excess Notation

1. 01001110

2. 00101101

Two's Complement

1. 11100110

2. 10001010

Two's Complement Binary Additions

1. No, Yes, Yes, 3

2. Yes, Yes, No, 1

3. No, No, Yes, 1

4. No, Yes, Yes, 3

Decimal to Excess 50 Notation

1. 65

2. 12

3. 97

4. 0

Conversion of Decimal Numbers to SEEZMMMM Format

1. 15512658

2. 04954320

3. 15310568

4. 15415428

Conversion of SEEZMMMM Format to Decimal Number

1. - 0.000067840

2. +176,830

3. -0.0000000075355

4. -44,387,000,000

Conversion of Decimal Exponent to Excess 127 Binary

1. $01110111_2 = 119_{10}$

2. $01111011_2 = 123_{10}$

3. $10001011_2 = 139_{10}$

Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. $+448_{10}$

2. -29_{10}

3. $+108_{10}$

4. $+51_{10}$

Class Exercises for Week 6

Recap on Signed Magnitude Notation

1. Represent the following decimal number in binary using Signed Magnitude Notation:

-97_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Binary Excess Notation to Decimal

1. What is the decimal number that is represented by 10101110_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

Recap on Decimal to Binary Excess Notation

1. What is the binary Excess Notation representation of the following decimal number:

-15_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: n = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

Recap on Two's Complement

1. Convert 74_{10} to binary then use Two's Complement to convert the unsigned binary representation of 74_{10} in to the Two's Complemented binary representation for -74_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Recap on Two's Complement Binary Additions

1. Add the following numbers together using two's complement binary representation and then answer the questions below:

67_{10} + -36_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

Recap on Decimal to Excess 50 Notation

1. Represent -34_{10} in excess 50 notation:

We have done this first one for you.

	+	50_{10}	=	

The correct answer is:

Recap on Conversion of Decimal Numbers to SEEZMMMM Format

1. Convert -435.679_{10} into the SEEZMMMM format:

We have done this first one for you.

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

Recap on Conversion of SEEZMMMM Format to Decimal Number

1. Convert 15322856 in the SEEZMMMM Format to a decimal number:

We have done this first one for you.

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

Recap on Conversion of Decimal Exponent to Excess 127 Binary

1. Represent an Exponent of -6_{10} in the Excess 127 format:

Excess	127	=								
Two's Complement Exponent	-6	=								
Result of addition of excess and exponent										
Carry's										

The correct answer is:

	=	

Recap on Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. Convert the following IEEE 754 single precision binary float to decimal:

$11000010110110110000000000000000_2$

We have done this one for you.

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

The Little Man Computer: First Program

Accessed at:

Look at the code below and complete the table description to explain what the code is doing at each line?

```
INP
OUT
HLT
```

Line	**Code**	**Description**
00	INP	
01	OUT	
02	HLT	

The Little Man Computer: Second Program

Look at the code below and complete the table description to explain what the code is doing at each line?

```
        INP
        STA   first
        INP
        STA   second
        LDA   first
        OUT
        LDA   second
        OUT
        HLT
first   DAT
second  DAT
```

Line	Code	Description
00	INP	
01	STA first	
02	INP	
03	STA second	
04	LDA first	
05	OUT	
06	LDA second	
07	OUT	
08	HLT	
09	first DAT	
10	second DAT	

The Little Man Computer: Third Program

Look at the code below and complete the table description to explain what the code is doing at each line?

```
        INP
        STA   first
        INP
        ADD   first
        OUT
        INP
        SUB   first
        OUT
        HLT
first   DAT
```

Line	Code	Description
00	INP	
01	STA first	
02	INP	
03	ADD first	
04	OUT	
05	INP	
06	SUB first	
07	OUT	
08	HLT	
09	first DAT	

The Little Man Computer: Fourth Program

Look at the code below and complete the table description to explain what the code is doing at each line?

```
          INP
          STA  first
          INP
          STA  second
          SUB  first
          BRP  secondbig
          LDA  first
          OUT
          BRA  stop
secondbig LDA  second
          OUT
stop      HLT
first     DAT
second    DAT
```

Line	Code	Description
00	INP	
01	STA first	
02	INP	
03	STA second	
04	SUB first	
05	BRP secondbig	
06	LDA first	
07	OUT	
08	BRA stop	
09	secondbig LDA second	
11	OUT	
11	stop HLT	
12	first DAT	
13	second DAT	

The Little Man Computer: Fifth Program

Look at the code below and complete the table description to explain what the code is doing at each line?

```
        LDA   one
        STA   count
        OUT
loop    LDA   count
        ADD   one
        OUT
        STA   count
        SUB   ten
        BRP   stop
        BRA   loop
stop    HLT
one     DAT   001
ten     DAT   010
count   DAT
```

Line	Code	Description
00	LDA one	
01	STA count	
02	OUT	
03	loop LDA count	
04	ADD one	
05	OUT	
06	STA count	
07	SUB ten	
08	BRP stop	
09	BRA loop	
10	stop HLT	
11	one DAT 001	
12	ten DAT 010	
13	count DAT	

The Little Man Computer: Programming Problems

1. Using the Little Man Computer design a program to multiply any number entered by a user by 10 and output the result.

Hint: There is no Multiplication command in the Little Man Computer, but you can Loop as many times as you want, so you will need to use Branching within your solution.

2. Using the Little Man Computer design a program to output how many fives will go exactly into any number entered by a user (you do not need to allow for any remainders).

Hint: There is no Divide command in the Little Man Computer, but you can Subtract as many times as you want.

The Little Man Computer: Further Challenging Programming Problems

Now that you have started to understand the Little Man Computer and its Assembly Language you should try developing programming solutions to some more challenging problems.

Develop programs in the Little Man Computer to:

1. Output Fibonacci numbers (a series of numbers in which each number (Fibonacci number) is the sum of the two preceding numbers. The simplest is the series 1, 1, 2, 3, 5, 8, etc.)
2. Output the first *n* Fibonacci numbers, where *n* is a user input
3. Take two number inputs *a, b* and compute *a* x *b*
4. Take a number input *a* and compute *a* divided by 2
5. Take two number inputs *a, b* and compute a divided by *b*
6. Take number user inputs until an input of 0 is received, then output the smallest of the inputs
7. Take two number inputs and output the highest common factor using Euclid's algorithm (an algorithm for finding the greatest common divisor of two numbers *a* and *b*)

The Answers for Week 6

Signed Magnitude Notation

1. 11100001_2

Binary Excess Notation to Decimal

1. $+46_{10}$

Decimal to Binary Excess Notation

1. 01110001_2

Two's Complement

1. 10110110_2

Two's Complement Binary Additions

1. No, Yes, Yes, 2

Decimal to Excess 50 Notation

1. $+16_{10}$

Conversion of Decimal Numbers to SEEZMMMM Format

1. 15343568

Conversion of SEEZMMMM Format to Decimal Number

1. -228.56_{10}

Conversion of Decimal Exponent to Excess 127 Binary

1. $01111001_2 = 121_{10}$

Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. -109.5_{10}

The Little Man Computer: Programming Problems

1.

```
            INP
            STA   value
loop        LDA   product
            ADD   value
            STA   product
            LDA   loopcount
            SUB   deductor
            STA   loopcount
            BRP   loop
            LDA   product
            OUT
            HLT
value       DAT
loopcount   DAT   009
product     DAT   000
deductor    DAT   001
```

2.

```
            INP
            STA   value
loop        LDA   count
            ADD   increment
            STA   count
            LDA   value
            SUB   divisor
            STA   value
            BRP   loop
            LDA   count
            SUB   increment
            OUT
            HLT
value       DAT
divisor     DAT   005
count       DAT   000
increment   DAT   001
```

Class Exercises for Week 7

Recap on Signed Magnitude Notation

1. Represent the following decimal number in binary using Signed Magnitude Notation:

$+107_{10}$

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Binary Excess Notation to Decimal

1. What is the decimal number that is represented by 10111011_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

Recap on Decimal to Binary Excess Notation

1. What is the binary Excess Notation representation of the following decimal number:

-63_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: n = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

Recap on Two's Complement

1. Convert 59_{10} to binary then use Two's Complement to convert the unsigned binary representation of 59_{10} in to the Two's Complemented binary representation for -59_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Recap on Two's Complement Binary Additions

1. Add the following numbers together using two's complement binary representation and then answer the questions below:

-43_{10} + -27_{10} =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

Recap on Decimal to Excess 50 Notation

1. Represent -16_{10} in excess 50 notation:

	+	50_{10}	=	

The correct answer is:

Recap on Conversion of Decimal Numbers to SEEZMMMM Format

1. Convert $+0.000548701_{10}$ into the SEEZMMMM format:

We have done this first one for you.

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

Recap on Conversion of SEEZMMMM Format to Decimal Number

1. Convert 04476823 in the SEEZMMMM Format to a decimal number:

We have done this first one for you.

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

Recap on Conversion of Decimal Exponent to Excess 127 Binary

1. Represent an Exponent of -23_{10} in the Excess 127 format:

Excess	127	=								
Two's Complement Exponent	-23	=								
Result of addition of excess and exponent										
Carry's										

The correct answer is:

	=	

Recap on Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. Convert the following IEEE 754 single precision binary float to decimal:

$01000011110110011000000000000000_2$

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

Recap on the Little Man Computer

1. Look at the Little Man Computer code below. What number will be in the Accumulator after the CPU executes the instruction held in memory location 4?

```
        LDA   first
        STA   second
        ADD   first
        SUB   second
        OUT
stop    HLT
first   DAT   010
second  DAT
```

The Register Transfer Language (RTL)

1. What does the following RTL notation indicate?

We have done this first one for you.

$[2_{(0:3)}] \leftarrow 1$

The RTL notation indicates that bits 0 to 3 of memory address 2 are to be set to one.

2. What does the following RTL notation indicate?

$[4] \leftarrow [8]$

3. What does the following RTL notation indicate?

$[8] \leftarrow [5]+1$

4. What does the following RTL notation indicate?

$[PC] \leftarrow [PC]+1$

MIPS Program One: Addition

The following MIPS Assembly Program was developed in the code editor MIPSter (http://www.downcastsystems.com/mipster/) and then run in the QtSpim MIPS Simulator (http://spimsimulator.sourceforge.net/)

```
# A Program to Add 10 to 10

        .data                   # Data declaration section

result: .asciiz "\n We Added 10 to 10 \n and the Result is: " #Memory location

        .text                   # Text declaration section

main:                           # Start of code section

        addi    $t0, $t0, 10    # Place immediate number 10 into t0 register

        addi    $t1, $t1, 10    # Place immediate number 10 into t1 register

        add     $t3, $t0, $t1   # Add the contents of t1 to t0 and place the result in t3

        li      $v0, 4          # Load immediate number 4 into v0 register (Print String)

        la      $a0, result     # Load memory address labelled 'result' and place into a0

        syscall                 # System call to carry out operation

        li      $v0, 1          # Load immediate number 1 into v0 register

        move    $a0, $t3        # Copy contents of register t3 to a0

        syscall                 # System call to carry out operation

        li      $v0, 10         # Load immediate number 10 into v0 register

        syscall                 # System call to carry out operation

# END OF PROGRAM
```

MIPS Program Two: Subtraction

```
# A Program to Subtract 10 from 20

        .data                   # Data declaration section
result: .asciiz "\n We Subtracted 10 from 20 \n and the Result is: "
        .text                   # Text declaration section
main:                           # Start of code section
        addi    $t0, $t0, 20
        addi    $t1, $t1, 10
        sub     $t3, $t0, $t1   # Subtract contents of t1 from t0 and place result in t3
        li      $v0, 4
        la      $a0, result
        syscall
        li      $v0, 1
        move    $a0, $t3
        syscall
        li      $v0, 10
        syscall
# END OF PROGRAM
```

MIPS Program Three: Multiplication

```
# A Program to Multiply 10 by 10

        .data                   # Data declaration section

result: .asciiz "\n We Multiply 10 by 10 \n and the Result is: "

        .text                   # Text declaration section

main:                           # Start of code section

        addi $t0, $t0, 10

        mult $t0, $t0           # Multiply t0 by t0

        mfhi $t1                # Move contents of Hi register to t1

        mflo $t2                # Move contents of Lo register to t2

        li $v0, 4

        la $a0, result

        syscall

        li $v0, 1

        move $a0, $t2

        syscall

        li $v0, 10

        syscall

# END OF PROGRAM
```

MIPS: Program Four: Division

```
# A Program to Divide 10 by 3
        .data                   # Data declaration section
result: .asciiz "\n We Divide 10 by 3 \n and the Result is: "
remain: .asciiz "\n Remainder: "
        .text                   # Text declaration section
main:                           # Start of code section
        addi $t0, $t0, 10
        addi $t1, $t1, 3
        div $t0, $t1            # Divide the contents of t0 by t1
        mfhi $t2
        mflo $t3
        li $v0, 4
        la $a0, result
        syscall
        li $v0, 1
        move $a0, $t3
        syscall
        li $v0, 4
        la $a0, remain
        syscall
        li $v0, 1
        move $a0, $t2
        syscall
        li $v0, 10
        syscall
# END OF PROGRAM
```

MIPS: Program Five: Branching Control

```
# A Program to Count Up to 10

        .data                   # Data declaration section

count:  .asciiz  "\n Look, I can count to 10: "

end:    .asciiz  "\n Well done me!"

new:    .asciiz  "\n"

        .text                   # Text declaration section

main:                           # Start of code section

        addi    $t0, $t0, 10

        addi    $t1, $t1, 0

        li      $v0, 4

        la      $a0, count

        syscall

        li      $v0, 4

        la      $a0, new

        syscall

loop:   addi    $t0, $t0, -1

        addi    $t1, $t1, 1

        li      $v0, 1

        move    $a0, $t1

        syscall

        li      $v0, 4

        la      $a0, new

        syscall

        blez    $t0, out        # Branch if t0 contents less than or equal to zero

        b       loop            # Branch always

out:    li      $v0, 4

        la      $a0, end

        syscall

        li      $v0, 10
```

```
        syscall
# END OF PROGRAM
```

MIPS: Program Six: Jumping

```
# A Program that Calls a Function
        .data                   # Data declaration section
enter:  .asciiz "\n Please Enter a Number: "
result: .asciiz "\n The result of adding the two numbers together is: "
        .text                   # Text declaration section
main:                           # Start of code section
        jal userent             # Jump to address labelled 'userent'
        move $t1, $t0
        jal userent
        move $t2, $t0
        add $t3, $t1, $t2
        li $v0, 4
        la $a0, result
        syscall
        li $v0, 1
        move $a0, $t3
        syscall
        li $v0, 10
        syscall
userent:
        li $v0, 4
        la $a0, enter
        syscall
        li $v0, 5
        syscall
        move $t0, $v0
        jr $ra                  # Jump to contents of return address register
```

```
# END OF PROGRAM
```

The Answers for Week 7

Signed Magnitude Notation

1. 01101011_2

Binary Excess Notation to Decimal

1. $+59_{10}$

Decimal to Binary Excess Notation

1. 01000001_2

Two's Complement

1. 11000101_2

Two's Complement Binary Additions

1. No, Yes, Yes, 4

Decimal to Excess 50 Notation

1. $+34_{10}$

Conversion of Decimal Numbers to SEEZMMMM Format

1. 04754870

Conversion of SEEZMMMM Format to Decimal Number

1. $+0.00000076823_{10}$

Conversion of Decimal Exponent to Excess 127 Binary

1. $01101000_2 = 104_{10}$

Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. $+435_{10}$

Recap on the Little Man Computer

1. 10

The Register Transfer Language

1. The RTL notation indicates that bits 0 to 3 of memory address 2 are to be set to one.

2. The RTL notation indicates that the contents of memory address 8 are transferred (copied) to memory address 4

3. The RTL notation indicates that the contents of memory address 5 are incremented by 1 and the result placed in to memory address 8

4. The RTL notation indicates that the Program Counter (PC) register will be incremented by 1.

Class Exercises for Week 8

Recap on Signed Magnitude Notation

1. Represent the following decimal number in binary using Signed Magnitude Notation:

-83_{10}

128	64	32	16	8	4	2	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

The correct answer is:

Recap on Binary Excess Notation to Decimal

1. What is the decimal number that is represented by 10101101_2 in Excess Notation?

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Unsigned binary to decimal conversion using positional notation								
Unsigned decimal value minus Excess ($2^{(n-1)}$) Note: *n* = number of bits								

The correct answer is:

Recap on Decimal to Binary Excess Notation

1. What is the binary Excess Notation representation of the following decimal number:

-35_{10}

Decimal plus Excess ($2^{(n-1)}$) Note: n = number of bits								
	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Convert Decimal with Excess to binary using positional notation								

The correct answer is:

Recap on Two's Complement

1. Convert 23_{10} to binary then use Two's Complement to convert the unsigned binary representation of 23_{10} in to the Two's Complemented binary representation for -23_{10}, what is the correct answer:

	128	64	32	16	8	4	2	1
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Positional notation used to convert decimal to binary								
Flipped bits								
One to add to the flipped bits above								
Result of addition of flipped bits and one								
Carry Bits								

The correct answer is:

Recap on Two's Complement Binary Additions

1. Add the following numbers together using two's complement binary representation and then answer the questions below:

$-15_{10} + 25_{10}$ =

			128	64	32	16	8	4	2	1
			2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+										

Did the calculation produce an overflow? YES / NO

Did the calculation produce a carryout? YES / NO

Would the calculation produce a correct result in an 8 bit system? YES / NO

How many bits were carried to the left during the calculation?

Recap on Decimal to Excess 50 Notation

1. Represent -6_{10} in excess 50 notation:

	+	50_{10}	=	

The correct answer is:

Recap on Conversion of Decimal Numbers to SEEZMMMM Format

1. Convert -0.000001113251_{10} into the SEEZMMMM format:

S = signed magnitude notation	=	
EE = Exponent Excess 50 notation	=	
ZMMMM = Mantissa	=	
Normalise Mantissa	=	

The correct answer is:

Recap on Conversion of SEEZMMMM Format to Decimal Number

1. Convert 15413283 in the SEEZMMMM Format to a decimal number:

S = signed magnitude notation	=		=	
EE = Exponent Excess 50 notation	=		=	
ZMMMM = Mantissa	=		=	

The correct answer is:

Recap on Conversion of Decimal Exponent to Excess 127 Binary

1. Represent an Exponent of -8_{10} in the Excess 127 format:

Excess	127	=								
Two's Complement Exponent	-8	=								
Result of addition of excess and exponent										
Carry's										

The correct answer is:

	=	

Recap on Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. Convert the following IEEE 754 single precision binary float to decimal:

$11000010011110010000000000000000_2$

Sign		=								
Excess		=								
Convert excess binary unsigned to decimal and subtract excess 127 to get exponent		=								

Mantissa	
Shift decimal point with exponent	
Convert shifted binary mantissa to decimal	
Add sign to converted decimal	

The correct answer is:

Recap on the Little Man Computer

1. Look at the Little Man Computer code below. What number will be in the Accumulator after the CPU executes the instruction held in memory location 6?

```
        LDA   first
        STA   second
        ADD   first
        STA   second
        LDA   first
        SUB   second
        OUT
stop          HLT
first         DAT   010
second        DAT
```

Recap on the Register Transfer Language (RTL)

1. What does the following RTL notation indicate?

We have done this first one for you.

$[7_{(1:8)}] \leftarrow 0$

The RTL notation indicates that bits 1 to 8 of memory address 7 are to be set to zero.

2. What does the following RTL notation indicate?

[2] = [6]

3. What does the following RTL notation indicate?

$[9] \leftarrow [3]-1$

4. What does the following RTL notation indicate?

$[PC] \leftarrow [8]+1$

Recap on MIPS

1. What does the following MIPS assembly code do?

We have done this first one for you.

```
addi    $t0, $t0, 10
```

This line of MIPS assembly code places immediate number 10 into register location t0

2. What does the following MIPS assembly code do?

```
li      $v0, 4
la      $a0, data
        syscall
```

3. What does the following MIPS assembly code do?

```
sub     $t3, $t0, $t1
```

4. What does the following MIPS assembly code do?

```
mult    $t0, $t0
```

5. What does the following MIPS assembly code do?

```
blez    $t0, out
```

6. What does the following MIPS assembly code do?

```
jal dothisnow
```

Huffman Coding

1. What is the Huffman Coding for the following alphabet:

A = {a/20, b/15, c/5, d/15, e/45}

c/5	d/15	b/15	a/20	e/45
0	1	0	1	
A1/20(c,d)		A2/35(b,a)		
0		1		
A3/55(c,d,b,a)				
1				0
A4/100(c,d,b,a,e)				

Letter		Huffman Coding
a	=	111
b	=	110
c	=	100
d	=	101
e	=	0

2. What is the Huffman Coding for the following alphabet:

A = {a/30, b/5, c/15, d/10, e/25}

Letter		Huffman Coding
a	=	
b	=	
c	=	
d	=	
e	=	

The Answers

Signed Magnitude Notation

1. 11010011_2

Binary Excess Notation to Decimal

1. $+45_{10}$

Decimal to Binary Excess Notation

1. 01011101_2

Two's Complement

1. 11101001_2

Two's Complement Binary Additions

1. No, Yes, Yes, 5

Decimal to Excess 50 Notation

1. $+44_{10}$

Conversion of Decimal Numbers to SEEZMMMM Format

1. 14511133

Conversion of SEEZMMMM Format to Decimal Number

1. -1328.3_{10}

Conversion of Decimal Exponent to Excess 127 Binary

1. $01110111_2 = 119_{10}$

Conversion of IEEE 754 Single Precision Binary Float to Decimal

1. -62.25_{10}

Recap on the Little Man Computer

1. -10

The Register Transfer Language

1. The RTL notation indicates that bits 1 to 8 of memory address 7 are to be set to zero.

2. The RTL notation indicates that the contents of memory address 2 are the same as memory address 6

3. The RTL notation indicates that the contents of memory address 3 are decremented by 1 and the result placed in to memory address 9

4. The RTL notation indicates that the contents of memory address 8 will be incremented by 1 and the result placed in to the Program Counter.

Recap on MIPS

1. This line of MIPS assembly code places immediate number 10 into register location t0

2. This line of MIPS assembly code prints a string located in a memory location address labelled as 'data'

3. This line of MIPS assembly code subtracts the contents of register t1 from register t0 and puts the result in register t3

4. This line of MIPS assembly code multiplies the contents of register t0 by register t0

5. This line of MIPS assembly code jumps the program counter to the line labelled 'out' if the contents of register t0 are less than or equal to zero (TRUE)

6. This line of MIPS assembly code jumps the program counter to the line labelled 'dothisnow' and places the next address in the return address register

Huffman Coding

2. A = {a/30, b/5, c/15, d/10, e/25}

b/5	d/10	c/15	e/25	a/30
0	1	0	1	
A1/15(b,d)		A2/40(c,e)		
0				1
A3/45(b,d,a)				
1		0		
A4/85(b,d,a,c,e)				

Letter		Huffman Coding
a	=	11
b	=	100
c	=	00
d	=	101
e	=	01

Class Exercises for Week 9

Truth Tables

1. Create a truth table for two inputs (*a, b*)

We have done this first one for you.

a	***b***
0	0
1	0
0	1
1	1

2. Create a truth table for four inputs (*a, b, c, d*):

a	***b***	***c***	***d***

Truth Tables from Logic Expressions

1. Create a Truth Table for the following logic expression:

$f = a \cdot b$

We have done this first one for you.

a	***b***	$\boldsymbol{a \cdot b}$	***f***
0	0	0	0
1	0	0	0
0	1	0	0
1	1	1	1

2. Create a Truth Table for the following logic expression:

$f = (a \cdot b) \cdot (c \cdot d)$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{d}$	$\boldsymbol{a \cdot b}$	$\boldsymbol{c \cdot d}$	$\boldsymbol{f}$

3. Create a Truth Table for the following logic expression:

$f = a + b$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{a + b}$	$\boldsymbol{f}$

4. Create a Truth Table for the following logic expression:

$f = (a + b) \cdot (c + d)$

a	b	c	d	$a + b$	$c + d$	f

5. Create a Truth Table for the following logic expression:

$f = \bar{a}$

a	$\bar{a}$	f

6. Create a Truth Table for the following logic expression:

$f = \bar{a} \cdot \bar{b}$

a	b	$\bar{a}$	$\bar{b}$	$\bar{a} \cdot \bar{b}$	f

7. Create a Truth Table for the following logic expression:

$f = \overline{a \cdot b}$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{a \cdot b}$	$\overline{\boldsymbol{a \cdot b}}$	$\boldsymbol{f}$

8. Create a Truth Table for the following logic expression:

$f = \overline{a + b}$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{a + b}$	$\overline{\boldsymbol{a + b}}$	$\boldsymbol{f}$

9. Create a Truth Table for the following logic expression:

$f = (\overline{a \cdot b}) \cdot (\overline{c \cdot d})$

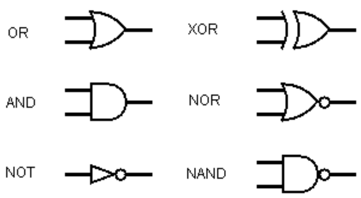
$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{d}$	$\boldsymbol{a \cdot b}$	$\boldsymbol{c \cdot d}$	$\overline{\boldsymbol{a \cdot b}}$	$\overline{\boldsymbol{c \cdot d}}$	$\boldsymbol{f}$

10. Create a Truth Table for the following logic expression:

$f = (\overline{a+b}) + (\overline{c+d})$

a	b	c	d	$a+b$	$c+d$	$\overline{a+b}$	$\overline{c+d}$	f

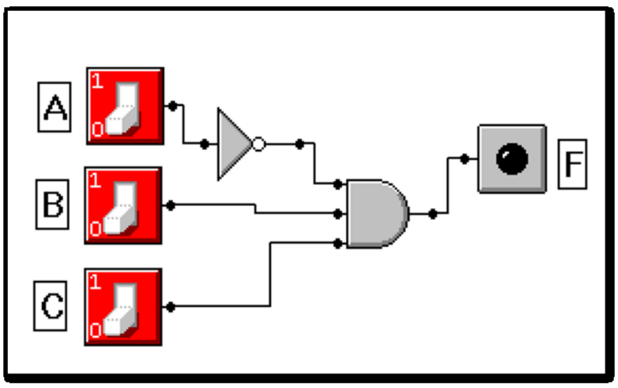
Building Logic Circuits from Expressions



1. Create a Logic Circuit Design for the following logic expression:

$f = \bar{a} \cdot b \cdot c$

We have done this first one for you.



2. Create a Logic Circuit Design for the following logic expression:

$f = (a + b) \cdot c$

3. Create a Logic Circuit Design for the following logic expression:

$f = (\overline{a + b}) \cdot (\overline{c \cdot d})$

4. Create a Logic Circuit Design for the following logic expression:

$f = a + (b \cdot c) + d$

The Answers for Week 9

Truth Tables

1.

a	***b***
0	0
1	0
0	1
1	1

2.

a	***b***	***c***	***d***
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	0
0	0	0	1
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	1

Truth Tables from Logic Expressions

1. $f = a \cdot b$

a	***b***	$\boldsymbol{a \cdot b}$	$\boldsymbol{f}$
0	0	0	0
1	0	0	0
0	1	0	0
1	1	1	1

2. $f = (a \cdot b) \cdot (c \cdot d)$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{d}$	$\boldsymbol{a \cdot b}$	$\boldsymbol{c \cdot d}$	$\boldsymbol{f}$
0	0	0	0	0	0	0
1	0	0	0	0	0	0
0	1	0	0	0	0	0
1	1	0	0	1	0	0
0	0	1	0	0	0	0
1	0	1	0	0	0	0
0	1	1	0	0	0	0
1	1	1	0	1	0	0
0	0	0	1	0	0	0
1	0	0	1	0	0	0
0	1	0	1	0	0	0
1	1	0	1	1	0	0
0	0	1	1	0	1	0
1	0	1	1	0	1	0
0	1	1	1	0	1	0
1	1	1	1	1	1	1

3. $f = a + b$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{a + b}$	$\boldsymbol{f}$
0	0	0	0
1	0	1	1
0	1	1	1
1	1	1	1

4. $f = (a + b) \cdot (c + d)$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{d}$	$\boldsymbol{a + b}$	$\boldsymbol{c + d}$	$\boldsymbol{f}$
0	0	0	0	0	0	0
1	0	0	0	1	0	0
0	1	0	0	1	0	0
1	1	0	0	1	0	0
0	0	1	0	0	1	0
1	0	1	0	1	1	1
0	1	1	0	1	1	1
1	1	1	0	1	1	1
0	0	0	1	0	1	0
1	0	0	1	1	1	1
0	1	0	1	1	1	1
1	1	0	1	1	1	1
0	0	1	1	0	1	0
1	0	1	1	1	1	1
0	1	1	1	1	1	1
1	1	1	1	1	1	1

5. $f = \bar{a}$

$\boldsymbol{a}$	$\boldsymbol{\bar{a}}$	f
0	1	1
1	0	0

6. $f = \bar{a} \cdot \bar{b}$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{\bar{a}}$	$\boldsymbol{\bar{b}}$	$\boldsymbol{f}$
0	0	1	1	1
1	0	0	1	0
0	1	1	0	0
1	1	0	0	0

7. $f = \overline{a \cdot b}$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{\overline{a \cdot b}}$	$\boldsymbol{f}$
0	0	1	1
1	0	1	1
0	1	1	1
1	1	0	0

8. $f = \overline{a + b}$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{\overline{a + b}}$	$\boldsymbol{f}$
0	0	1	1
1	0	0	0
0	1	0	0
1	1	0	0

9. $f = (\overline{a \cdot b}) \cdot (\overline{c \cdot d})$

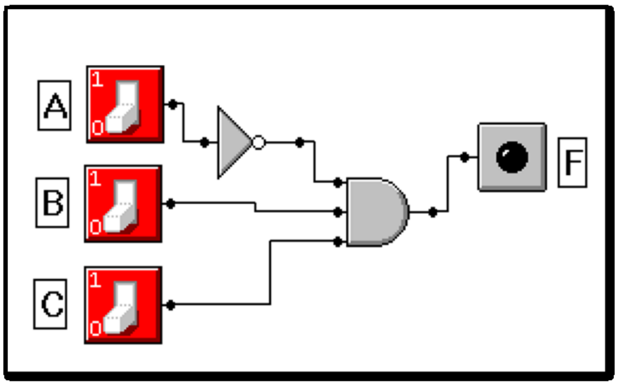
a	b	c	d	$\overline{a \cdot b}$	$\overline{c \cdot d}$	f
0	0	0	0	1	1	1
1	0	0	0	1	1	1
0	1	0	0	1	1	1
1	1	0	0	0	1	0
0	0	1	0	1	1	1
1	0	1	0	1	1	1
0	1	1	0	1	1	1
1	1	1	0	0	1	0
0	0	0	1	1	1	1
1	0	0	1	1	1	1
0	1	0	1	1	1	1
1	1	0	1	0	1	0
0	0	1	1	1	0	0
1	0	1	1	1	0	0
0	1	1	1	1	0	0
1	1	1	1	0	0	0

10. $f = \left(\overline{a + b}\right) + (\overline{c + d})$

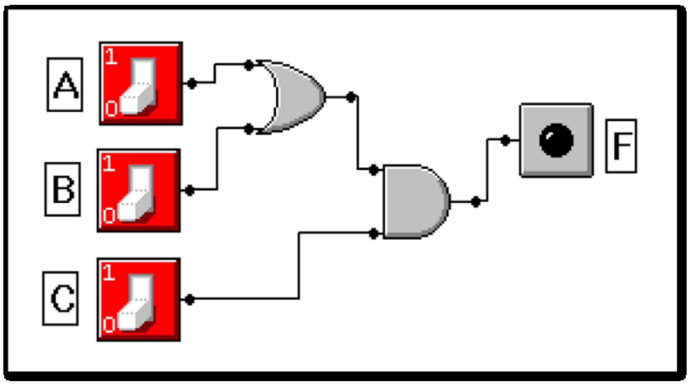
a	b	c	d	$\overline{a + b}$	$\overline{c + d}$	f
0	0	0	0	1	1	1
1	0	0	0	0	1	1
0	1	0	0	0	1	1
1	1	0	0	0	1	1
0	0	1	0	1	0	1
1	0	1	0	0	0	0
0	1	1	0	0	0	0
1	1	1	0	0	0	0
0	0	0	1	1	0	1
1	0	0	1	0	0	0
0	1	0	1	0	0	0
1	1	0	1	0	0	0
0	0	1	1	1	0	1
1	0	1	1	0	0	0
0	1	1	1	0	0	0
1	1	1	1	0	0	0

Building Logic Circuits from Expressions

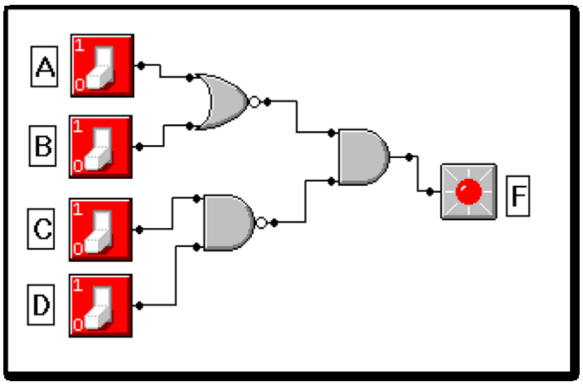
1. $f = \bar{a} \cdot b \cdot c$



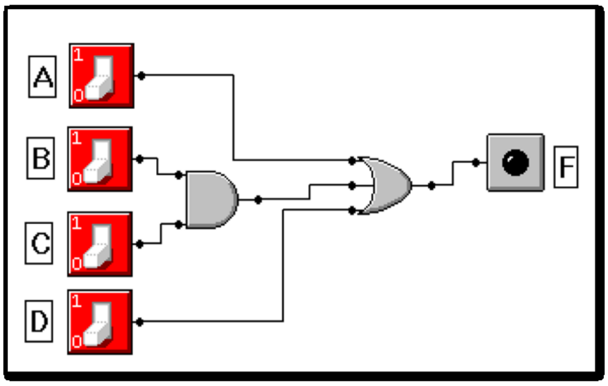
2. $f = (a + b) \cdot c$



3. $f = (\overline{a + b}) \cdot (\overline{c \cdot d})$



4. $f = a + (b \cdot c) + d$



Class Exercises for Week 10

Recap on Truth Tables

1. Create a Truth Table for the following logic expression:

$f = \overline{b} \cdot (a \cdot b) + (\overline{c \cdot d})$

a	***b***	***c***	***d***	$\overline{b}$	$a \cdot b$	$\overline{b} \cdot (a \cdot b)$	$c \cdot d$	$\overline{c \cdot d}$	***f***

2. Create a Truth Table for the following logic expression:

$f = (\overline{\overline{a} + b}) \cdot (c + \overline{d})$

a	***b***	***c***	***d***	$\overline{a}$	$\overline{a} + b$	$\overline{\overline{a} + b}$	$\overline{d}$	$c + \overline{d}$	***f***

Recap on Building Logic Circuits from Expressions

1. Create a Logic Circuit Design for the following logic expression:

$f = (a + b) \cdot c$

2. Create a Logic Circuit Design for the following logic expression:

$f = (\overline{a + b}) \cdot (\overline{c \cdot d})$

3. Create a Logic Circuit Design for the following logic expression:

$f = a \cdot (\overline{\bar{c} + d}) \cdot b$

Sum of Products (SoP) Representations

1. Create a Logic Circuit Design for the following SoP logic expression:

$f = (a \cdot b) + (c \cdot d)$

2. Create a Logic Circuit Design for the following SoP logic expression:

$f = (\bar{a} \cdot b) + (c \cdot \bar{d})$

3. Create a Logic Circuit Design for the following SoP logic expression:

$f = (\overline{a \cdot b}) + (\overline{c \cdot \bar{d}})$

4. Create a Logic Circuit Design for the following SoP logic expression:

$f = \overline{(a \cdot b) + (c \cdot d)}$

Product of Sums (PoS) Representations

1. Create a Logic Circuit Design for the following PoS logic expression:

$f = (a + b) \cdot (c + d)$

2. Create a Logic Circuit Design for the following SoP logic expression:

$f = (\bar{a} + b) \cdot (c + \bar{d})$

3. Create a Logic Circuit Design for the following SoP logic expression:

$f = (\overline{a + b}) \cdot (\overline{c + \bar{d}})$

4. Create a Logic Circuit Design for the following SoP logic expression:

$f = \overline{(a + b) \cdot (c + d)}$

Specified Sum of Products Expressions

1. Create a Logic Circuit Design for the following specified SoP logic expression:

$f = \left(\bar{a} \cdot \bar{b} \cdot c\right) + (a \cdot b \cdot \bar{c})$

Specified Product of Sums Expressions

1. Create a Logic Circuit Design for the following specified PoS logic expression:

$f = (a + \bar{b} + c) \cdot (a + \bar{b} + \bar{c})$

SoP Expressions and Circuit Design from a Specification

1. What is the fully defined SoP expression for the following specification truth table?

a	*b*	*c*	*f*	*Minterms*
0	0	0	1	
1	0	0	1	
0	1	0	0	
1	1	0	0	
0	0	1	0	
1	0	1	0	
0	1	1	0	
1	1	1	1	

The correct fully defined SoP expression is:

The Circuit Design is:

PoS Expressions and Circuit Design from a Specification

1. What is the fully defined PoS expression and Circuit Design for the following specification truth table?

a	***b***	***c***	***f***	***Maxterms***
0	0	0	1	
1	0	0	0	
0	1	0	1	
1	1	0	1	
0	0	1	1	
1	0	1	1	
0	1	1	0	
1	1	1	1	

The correct fully defined PoS expression is:

The Circuit Design is:

Specified Truth Table from a SoP Expression

1. What was the specified truth table (*f*) that was used to develop the following SoP expression?

$$f = (a \cdot b \cdot c) + (\bar{a} \cdot b \cdot \bar{c}) + (\bar{a} \cdot \bar{b} \cdot c)$$

a	***b***	***c***	***f***
0	0	0	
1	0	0	
0	1	0	
1	1	0	
0	0	1	
1	0	1	
0	1	1	
1	1	1	

Specified Truth Table from a PoS Expression

1. What was the specified truth table (*f*) that was used to develop the following PoS expression?

$$f = \left(a + \bar{b} + c\right) \cdot (a + b + \bar{c}) \cdot (\bar{a} + \bar{b} + \bar{c})$$

a	***b***	***c***	***f***
0	0	0	
1	0	0	
0	1	0	
1	1	0	
0	0	1	
1	0	1	
0	1	1	
1	1	1	

The Answers for Week 10

Recap on Truth Tables

1.

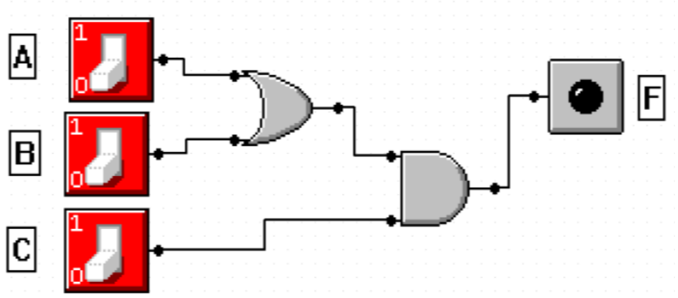
a	b	c	d	$\overline{b}$	$a \cdot b$	$\overline{b} \cdot (a \cdot b)$	$c \cdot d$	$\overline{c \cdot d}$	f
0	0	0	0	1	0	0	0	1	1
1	0	0	0	1	0	0	0	1	1
0	1	0	0	0	0	0	0	1	1
1	1	0	0	0	1	0	0	1	1
0	0	1	0	1	0	0	0	1	1
1	0	1	0	1	0	0	0	1	1
0	1	1	0	0	0	0	0	1	1
1	1	1	0	0	1	0	0	1	1
0	0	0	1	1	0	0	0	1	1
1	0	0	1	1	0	0	0	1	1
0	1	0	1	0	0	0	0	1	1
1	1	0	1	0	1	0	0	1	1
0	0	1	1	1	0	0	1	0	0
1	0	1	1	1	0	0	1	0	0
0	1	1	1	0	0	0	1	0	0
1	1	1	1	0	1	0	1	0	0

2.

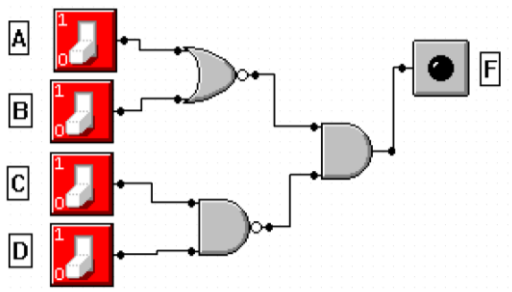
a	b	c	d	$\overline{a}$	$\overline{a} + b$	$\overline{\overline{a} + b}$	$\overline{d}$	$c + \overline{d}$	f
0	0	0	0	1	1	0	1	1	0
1	0	0	0	0	0	1	1	1	1
0	1	0	0	1	1	0	1	1	0
1	1	0	0	0	1	0	1	1	0
0	0	1	0	1	1	0	1	1	0
1	0	1	0	0	0	1	1	1	1
0	1	1	0	1	1	0	1	1	0
1	1	1	0	0	1	0	1	1	0
0	0	0	1	1	1	0	0	0	0
1	0	0	1	0	0	1	0	0	0
0	1	0	1	1	1	0	0	0	0
1	1	0	1	0	1	0	0	0	0
0	0	1	1	1	1	0	0	1	0
1	0	1	1	0	0	1	0	1	1
0	1	1	1	1	1	0	0	1	0
1	1	1	1	0	1	0	0	1	0

Recap on Building Logic Circuits from Expressions

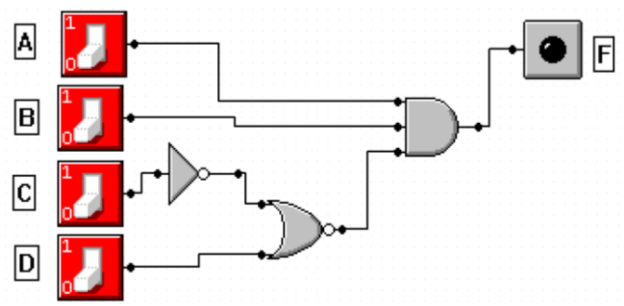
1. $f = (a + b) \cdot c$



2. $f = (\overline{a + b}) \cdot (\overline{c \cdot d})$

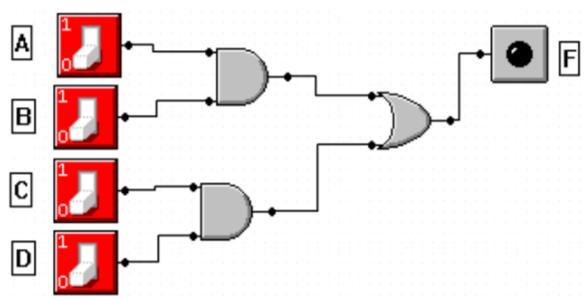


3. $f = a \cdot (\overline{\bar{c} + d}) \cdot b$

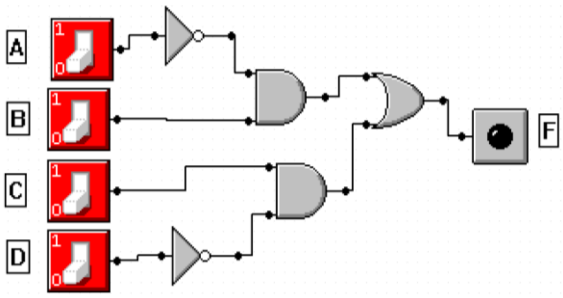


Sum of Products (SoP) Representations

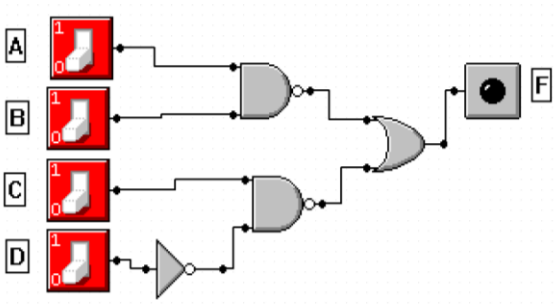
1. $f = (a \cdot b) + (c \cdot d)$



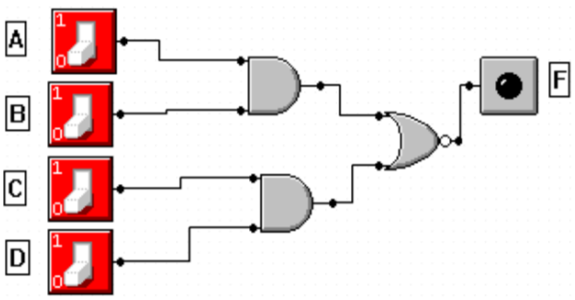
2. $f = (\bar{a} \cdot b) + (c \cdot \bar{d})$



3. $f = (\overline{a \cdot b}) + (\overline{c \cdot \bar{d}})$

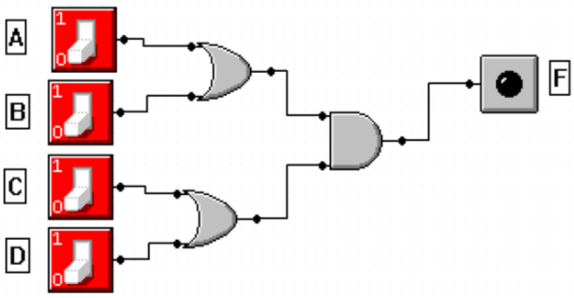


4. $f = \overline{(a \cdot b) + (c \cdot d)}$

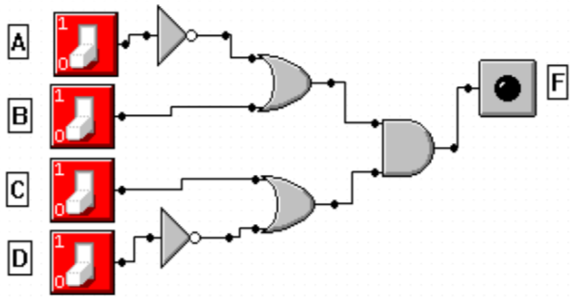


Product of Sums (PoS) Representations

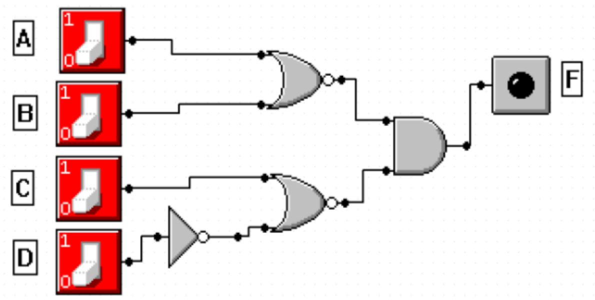
1. $f = (a + b) \cdot (c + d)$



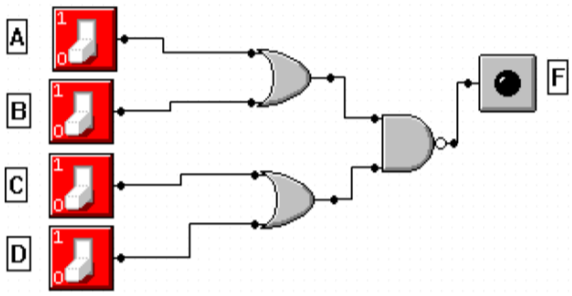
2. $f = (\bar{a} + b) \cdot (c + \bar{d})$



3. $f = (\overline{a + b}) \cdot (\overline{c + \bar{d}})$

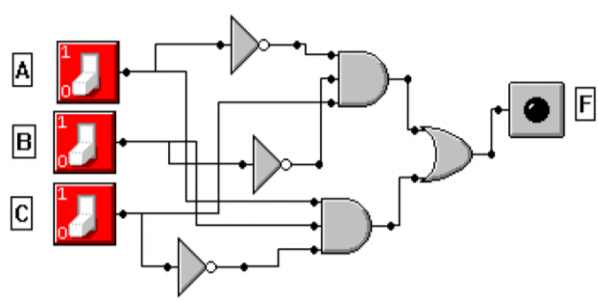


4. $f = \overline{(a + b) \cdot (c + d)}$



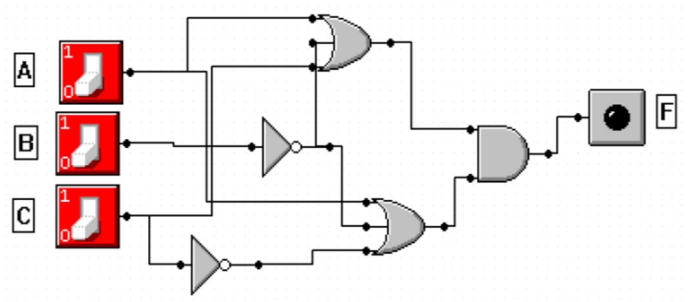
Specified Sum of Products Expressions

1. $f = \left(\bar{a} \cdot \bar{b} \cdot c\right) + (a \cdot b \cdot \bar{c})$



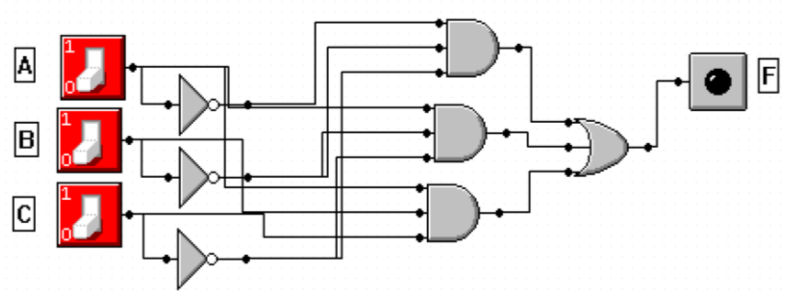
Specified Product of Sums Expressions

1. $f = (a + \bar{b} + c) \cdot (a + \bar{b} + \bar{c})$



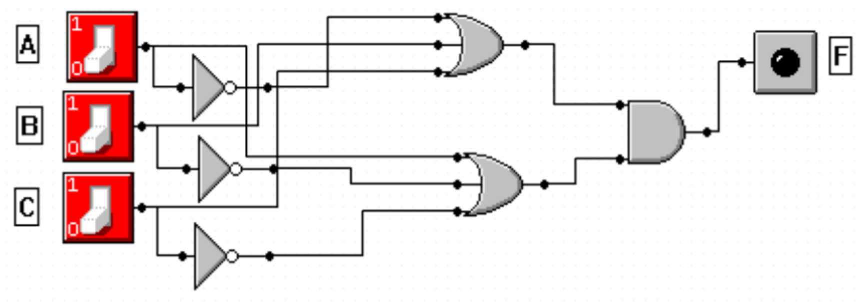
SoP Expressions and Circuit Design from a Specification

1. $f = (\bar{a} \cdot \bar{b} \cdot \bar{c}) + (a \cdot \bar{b} \cdot \bar{c}) + (a \cdot b \cdot c)$



PoS Expressions and Circuit Design from a Specification

1. $f = (\bar{a} + b + c) \cdot (a + \bar{b} + \bar{c})$



Specified Truth Table from a SoP Expression

1. $f = (a \cdot b \cdot c) + (\bar{a} \cdot b \cdot \bar{c}) + (\bar{a} \cdot \bar{b} \cdot c)$

a	***b***	***c***	***f***
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Specified Truth Table from a PoS Expression

1. $f = \left(a + \bar{b} + c\right) \cdot (a + b + \bar{c}) \cdot (\bar{a} + \bar{b} + \bar{c})$

a	***b***	***c***	***f***
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Class Exercises for Week 11

Recap on Truth Tables

1. Create a Truth Table for the following logic expression:

$f = \overline{b} \cdot (a \cdot b) + (\overline{c \cdot d})$

$\boldsymbol{a}$	$\boldsymbol{b}$	$\boldsymbol{c}$	$\boldsymbol{d}$	$\boldsymbol{\overline{b}}$	$\boldsymbol{a \cdot b}$	$\boldsymbol{\overline{b} \cdot (a \cdot b)}$	$\boldsymbol{c \cdot d}$	$\boldsymbol{\overline{c \cdot d}}$	$\boldsymbol{f}$

Recap on Building Logic Circuits from Expressions

1. Create a Logic Circuit Design for the following logic expression:

$f = (a + \overline{b} + \overline{c}) \cdot d$

Recap on SoP and Pos Expressions and Circuit Design from a Specification

1. What is the fully defined SoP expression for the following specification truth table?

a	*b*	*c*	*f*	*Minterms*
0	0	0	1	
1	0	0	1	
0	1	0	0	
1	1	0	0	
0	0	1	0	
1	0	1	0	
0	1	1	0	
1	1	1	1	

The correct fully defined SoP expression is:

The Circuit Design is:

2. What is the fully defined PoS expression and Circuit Design for the following specification truth table?

a	*b*	*c*	*f*	***Maxterms***
0	0	0	1	
1	0	0	0	
0	1	0	1	
1	1	0	1	
0	0	1	1	
1	0	1	1	
0	1	1	0	
1	1	1	1	

The correct fully defined PoS expression is:

The Circuit Design is:

Recap on Specified Truth Tables from SoP and PoS Expressions

1. What was the specified truth table (*f*) that was used to develop the following SoP expression?

$$f = (a \cdot b \cdot c) + (\bar{a} \cdot b \cdot \bar{c}) + (\bar{a} \cdot \bar{b} \cdot c)$$

a	***b***	***c***	***f***
0	0	0	
1	0	0	
0	1	0	
1	1	0	
0	0	1	
1	0	1	
0	1	1	
1	1	1	

2. What was the specified truth table (*f*) that was used to develop the following PoS expression?

$$f = \left(a + \bar{b} + c\right) \cdot (a + b + \bar{c}) \cdot (\bar{a} + \bar{b} + \bar{c})$$

a	***b***	***c***	***f***
0	0	0	
1	0	0	
0	1	0	
1	1	0	
0	0	1	
1	0	1	
0	1	1	
1	1	1	

Drawing out a SoP or PoS Karnaugh Map (K-map)

1. Draw a two variable K-map in the SoP form with inputs labelled as: ***a b***

Then enter the correct binary/hex number, created by the inputs, for each minterm.

We have done this first one for you (red numbers are binary and blue are hex for each minterm).

SoP K-map	$\overline{b}$	b
$\overline{a}$	00 0	01 1
a	10 2	11 3

2. Draw a three variable K-map in the SoP form with inputs labelled as: ***a b c***

Then enter the correct binary/hex number, created by the inputs, for each minterm.

SoP K-map				

3. Draw a four variable K-map in the SoP form with inputs labelled as: ***a b c d***

Then enter the correct binary/hex number, created by the inputs, for each minterm.

SoP K-map				

4. Draw a two variable K-map in the PoS form with inputs labelled as: ***a b***

Then enter the correct binary/hex number, created by the inputs, for each maxterm.

PoS K-map		

5. Draw a three variable K-map in the PoS form with inputs labelled as: ***a b c***

Then enter the correct binary/hex number, created by the inputs, for each maxterm.

PoS K-map				

6. Draw a four variable K-map in the PoS form with inputs labelled as: ***a b c d***

Then enter the correct binary/hex number, created by the inputs, for each maxterm.

PoS K-map				

Placing Minterms (SoP) or Maxterms (PoS) on a Karnaugh Map (K-map)

1. Look at the following SoP expression and then map its minterms (1's) to an appropriate K-map:

$f = (\bar{a} \cdot b \cdot c \cdot d) + (\bar{a} \cdot b \cdot c \cdot \bar{d}) + (a \cdot b \cdot c \cdot d) + (a \cdot b \cdot c \cdot \bar{d})$

SoP K-map				

2. Look at the following PoS expression and then map its maxterms (0's) to an appropriate K-map:

$$f = (\bar{a} + b + \bar{c} + d) \cdot (\bar{a} + b + \bar{c} + \bar{d}) \cdot (a + b + \bar{c} + d) \cdot (a + \bar{b} + c + \bar{d})$$

PoS K-map				

Karnaugh Map Coordinates to Determine SoP and PoS Expressions

1. Look at the following coordinates and assuming four input variables (***a b c d***) use them to determine the SoP and PoS expressions they represent

$f = \Sigma m(9BDF)$

PoS K-map				

SoP K-map				

The SoP expression from the coordinates provided is:

The PoS expression from the coordinates provided is:

2. Look at the following coordinates and assuming four input variables (***a b c d***) use them to determine the SoP and PoS expressions they represent

$f = \Sigma m(8E5A)$

PoS K-map				

SoP K-map				

The SoP expression from the coordinates provided is:

The PoS expression from the coordinates provided is:

Identifying Prime Implicants in Karnaugh Maps

1. Create K-maps for the following SoP and PoS expressions and then circle the prime implicants in each

$$f = (\bar{a} \cdot \mathrm{b} \cdot \bar{c} \cdot \bar{d}) + (\mathrm{a} \cdot \mathrm{b} \cdot \bar{c} \cdot \bar{d}) + (\mathrm{a} \cdot \bar{b} \cdot \bar{c} \cdot \bar{d}) + (\mathrm{a} \cdot \bar{b} \cdot \bar{c} \cdot \mathrm{d}) + (\bar{a} \cdot \mathrm{b} \cdot \mathrm{c} \cdot \bar{d})$$

$$f = (\mathrm{a} + \mathrm{b} + \mathrm{c} + \mathrm{d}) \cdot (\mathrm{a} + \mathrm{b} + \mathrm{c} + \bar{d}) \cdot (\mathrm{a} + \bar{b} + \mathrm{c} + \bar{d}) \cdot (\mathrm{a} + \bar{b} + \mathrm{c} + \mathrm{d}) \cdot (\mathrm{a} + \mathrm{b} + \bar{c} + \mathrm{d})$$

PoS K-map				

SoP K-map				

Simplifying SoP and PoS Expressions using Karnaugh Maps

1. Simplify the following SoP and PoS expressions using K-maps

$$f = (\mathrm{a} \cdot \mathrm{b} \cdot \mathrm{c} \cdot \mathrm{d}) + (\mathrm{a} \cdot \mathrm{b} \cdot \bar{c} \cdot \mathrm{d}) + (\mathrm{a} \cdot \bar{b} \cdot \bar{c} \cdot \mathrm{d}) + (\mathrm{a} \cdot \bar{b} \cdot \mathrm{c} \cdot \mathrm{d}) + (\bar{a} \cdot \bar{b} \cdot \mathrm{c} \cdot \mathrm{d})$$

$$f = (\mathrm{a} + \bar{b} + \mathrm{c} + \mathrm{d}) \cdot (\mathrm{a} + \bar{b} + \mathrm{c} + \bar{d}) \cdot (\mathrm{a} + \mathrm{b} + \mathrm{c} + \bar{d}) \cdot (\mathrm{a} + \mathrm{b} + \bar{c} + \bar{d}) \cdot (\bar{a} + \mathrm{b} + \bar{c} + \bar{d})$$

PoS K-map				

SoP K-map				

The simplified SoP expression is:

The simplified PoS expression is:

The Answers for Week 11

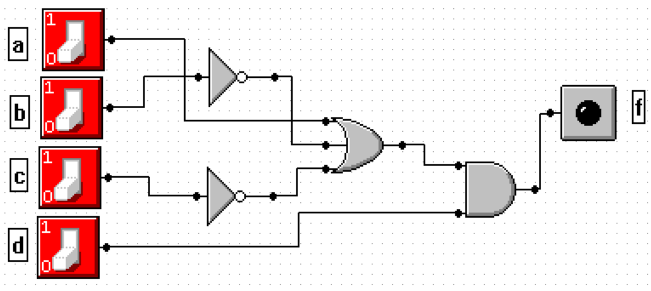
Recap on Truth Tables

1.

a	b	c	d	$\overline{b}$	$a \cdot b$	$\overline{b} \cdot (a \cdot b)$	$c \cdot d$	$\overline{c \cdot d}$	f
0	0	0	0	1	0	0	0	1	1
1	0	0	0	1	0	0	0	1	1
0	1	0	0	0	0	0	0	1	1
1	1	0	0	0	1	0	0	1	1
0	0	1	0	1	0	0	0	1	1
1	0	1	0	1	0	0	0	1	1
0	1	1	0	0	0	0	0	1	1
1	1	1	0	0	1	0	0	1	1
0	0	0	1	1	0	0	0	1	1
1	0	0	1	1	0	0	0	1	1
0	1	0	1	0	0	0	0	1	1
1	1	0	1	0	1	0	0	1	1
0	0	1	1	1	0	0	1	0	0
1	0	1	1	1	0	0	1	0	0
0	1	1	1	0	0	0	1	0	0
1	1	1	1	0	1	0	1	0	0

Recap on Building Logic Circuits from Expressions

1. $f = (a + \overline{b} + \overline{c}) \cdot d$

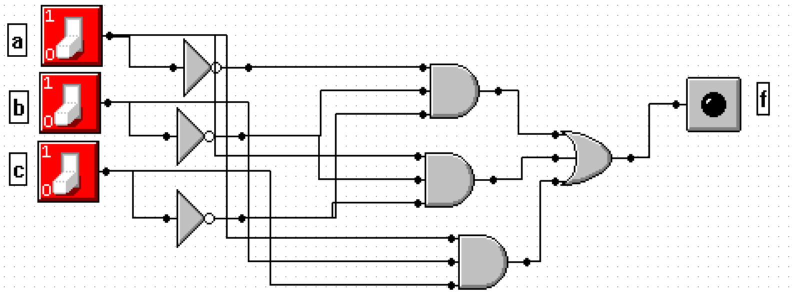


Recap on SoP and Pos Expressions and Circuit Design from a Specification

1.

a	*b*	*c*	*f*	*Minterms*
0	0	0	1	$(\bar{a} \cdot \bar{b} \cdot \bar{c})$
1	0	0	1	$(a \cdot \bar{b} \cdot \bar{c})$
0	1	0	0	
1	1	0	0	
0	0	1	0	
1	0	1	0	
0	1	1	0	
1	1	1	1	$(a \cdot b \cdot c)$

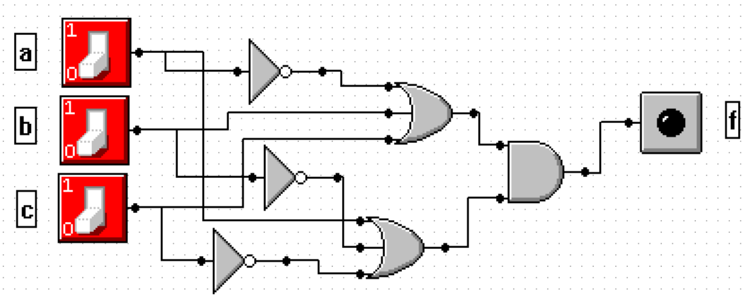
$$f = (\bar{a} \cdot \bar{b} \cdot \bar{c}) + (a \cdot \bar{b} \cdot \bar{c}) + (a \cdot b \cdot c)$$



2.

a	*b*	*c*	*f*	*Maxterms*
0	0	0	1	
1	0	0	0	$(\bar{a} + b + c)$
0	1	0	1	
1	1	0	1	
0	0	1	1	
1	0	1	1	
0	1	1	0	$(a + \bar{b} + \bar{c})$
1	1	1	1	

$$f = (\bar{a} + b + c) \cdot (a + \bar{b} + \bar{c})$$



Recap on Specified Truth Tables from SoP and PoS Expressions

1. $f = (a \cdot b \cdot c) + (\bar{a} \cdot b \cdot \bar{c}) + (\bar{a} \cdot \bar{b} \cdot c)$

a	***b***	***c***	***f***
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

2. $f = \left(a + \bar{b} + c\right) \cdot (a + b + \bar{c}) \cdot (\bar{a} + \bar{b} + \bar{c})$

a	***b***	***c***	***f***
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Drawing out a SoP or PoS Karnaugh Map (K-map)

1.

SoP K-map	$\bar{b}$	b
$\bar{a}$	**00** **0**	**01** **1**
a	**10** **2**	**11** **3**

2.

SoP K-map	$\overline{b} \cdot \overline{c}$	$\overline{b} \cdot c$	$b \cdot c$	$b \cdot \overline{c}$
$\overline{a}$	000 0	001 1	011 3	010 2
a	100 4	101 5	111 7	110 6

3.

SoP K-map	$\overline{c} \cdot \overline{d}$	$\overline{c} \cdot d$	$c \cdot d$	$c \cdot \overline{d}$
$\overline{a} \cdot \overline{b}$	0000 0	0001 1	0011 3	0010 2
$\overline{a} \cdot b$	0100 4	0101 5	0111 7	0110 6
$a \cdot b$	1100 C	1101 D	1111 F	1110 E
$a \cdot \overline{b}$	1000 8	1001 9	1011 B	1010 A

4.

PoS K-map	b	$\overline{b}$
a	00 0	01 1
$\overline{a}$	10 2	11 3

5.

PoS K-map	$\boldsymbol{b+c}$	$\boldsymbol{b+\bar{c}}$	$\boldsymbol{\bar{b}+\bar{c}}$	$\boldsymbol{\bar{b}+c}$
$\boldsymbol{a}$	000 0	001 1	011 3	010 2
$\boldsymbol{\bar{a}}$	100 4	101 5	111 7	110 6

6.

PoS K-map	$\boldsymbol{c+d}$	$\boldsymbol{c+\bar{d}}$	$\boldsymbol{\bar{c}+\bar{d}}$	$\boldsymbol{\bar{c}+d}$
$\boldsymbol{a+b}$	0000 0	001 1	011 3	010 2
$\boldsymbol{a+\bar{b}}$	0100 4	0101 5	0111 7	0110 6
$\boldsymbol{\bar{a}+\bar{b}}$	1100 C	1101 D	1111 F	1110 E
$\boldsymbol{\bar{a}+b}$	1000 8	1001 9	1011 B	1010 A

Placing Minterms (SoP) or Maxterms (PoS) on a Karnaugh Map (K-map)

1. $f = (\bar{a} \cdot b \cdot c \cdot d) + (\bar{a} \cdot b \cdot c \cdot \bar{d}) + (a \cdot b \cdot c \cdot d) + (a \cdot b \cdot c \cdot \bar{d})$

SoP K-map	$\bar{c} \cdot \bar{d}$	$\bar{c} \cdot d$	$c \cdot d$	$c \cdot \bar{d}$
$\bar{a} \cdot \bar{b}$	0	0	0	0
$\bar{a} \cdot b$	0	0	**1**	**1**
$a \cdot b$	0	0	**1**	**1**
$a \cdot \bar{b}$	0	0	0	0

2. $f = (\bar{a} + b + \bar{c} + d) \cdot (\bar{a} + b + \bar{c} + \bar{d}) \cdot (a + b + \bar{c} + d) \cdot (a + \bar{b} + c + \bar{d})$

PoS K-map	$c + d$	$c + \bar{d}$	$\bar{c} + \bar{d}$	$\bar{c} + d$
$a + b$	1	1	1	**0**
$a + \bar{b}$	1	**0**	1	1
$\bar{a} + \bar{b}$	1	1	1	1
$\bar{a} + b$	1	1	**0**	**0**

Karnaugh Map Coordinates to Determine SoP and PoS Expressions

1. $f = \Sigma m(9BDF)$

PoS K-map	$c + d$	$c + \bar{d}$	$\bar{c} + \bar{d}$	$\bar{c} + d$
$a + b$	1	1	1	1
$a + \bar{b}$	1	1	1	1
$\bar{a} + \bar{b}$	1	**0**	**0**	1
$\bar{a} + b$	1	**0**	**0**	1

SoP K-map	$\bar{c} \cdot \bar{d}$	$\bar{c} \cdot d$	$c \cdot d$	$c \cdot \bar{d}$
$\bar{a} \cdot \bar{b}$	0	0	0	0
$\bar{a} \cdot b$	0	0	0	0
$a \cdot b$	0	**1**	**1**	0
$a \cdot \bar{b}$	0	**1**	**1**	0

$$f = (\bar{a} + \bar{b} + c + \bar{d}) \cdot (\bar{a} + \bar{b} + \bar{c} + \bar{d}) \cdot (\bar{a} + b + c + \bar{d}) \cdot (\bar{a} + b + \bar{c} + \bar{d})$$

$$f = (a \cdot b \cdot \bar{c} \cdot d) + (a \cdot b \cdot c \cdot d) + \left(a \cdot \bar{b} \cdot \bar{c} \cdot d\right) + \left(a \cdot \bar{b} \cdot c \cdot d\right)$$

2. $f = \Sigma m(8E5A)$

PoS K-map	$c + d$	$c + \bar{d}$	$\bar{c} + \bar{d}$	$\bar{c} + d$
$a + b$	1	1	1	1
$a + \bar{b}$	1	**0**	1	1
$\bar{a} + \bar{b}$	1	1	1	**0**
$\bar{a} + b$	**0**	1	1	**0**

SoP K-map	$\bar{c} \cdot \bar{d}$	$\bar{c} \cdot d$	$c \cdot d$	$c \cdot \bar{d}$
$\bar{a} \cdot \bar{b}$	0	0	0	0
$\bar{a} \cdot b$	0	**1**	0	0
$a \cdot b$	0	0	0	**1**
$a \cdot \bar{b}$	**1**	0	0	**1**

$$f = (\bar{a} + b + c + d) \cdot (a + \bar{b} + c + \bar{d}) \cdot (\bar{a} + \bar{b} + \bar{c} + d) \cdot (\bar{a} + b + \bar{c} + d)$$

$$f = \left(a \cdot \bar{b} \cdot \bar{c} \cdot \bar{d}\right) + (\bar{a} \cdot b \cdot \bar{c} \cdot d) + \left(a \cdot b \cdot c \cdot \bar{d}\right) + \left(a \cdot \bar{b} \cdot c \cdot \bar{d}\right)$$

Identifying Prime Implicants in Karnaugh Maps

1.

$$f = (\bar{a}\cdot \mathrm{b}\cdot \bar{c}\cdot \bar{d}) + (\mathrm{a}\cdot \mathrm{b}\cdot \bar{c}\cdot \bar{d}) + (\mathrm{a}\cdot \bar{b}\cdot \bar{c}\cdot \bar{d}) + (\mathrm{a}\cdot \bar{b}\cdot \bar{c}\cdot \mathrm{d}) + (\bar{a}\cdot \mathrm{b}\cdot \mathrm{c}\cdot \bar{d})$$

$$f = (\mathrm{a}+\mathrm{b}+\mathrm{c}+\mathrm{d})\cdot(\mathrm{a}+\mathrm{b}+\mathrm{c}+\bar{d})\cdot(\mathrm{a}+\bar{b}+\mathrm{c}+\bar{d})\cdot(\mathrm{a}+\bar{b}+\mathrm{c}+\mathrm{d})\cdot(\mathrm{a}+\mathrm{b}+\bar{c}+\mathrm{d})$$

PoS K-map	$c+d$	$c+\bar{d}$	$\bar{c}+\bar{d}$	$\bar{c}+d$
$a+b$	**0**	**0**		**0**
$a+\bar{b}$	**0**	**0**		
$\bar{a}+\bar{b}$				
$\bar{a}+b$				

SoP K-map	$\bar{c}\cdot\bar{d}$	$\bar{c}\cdot d$	$c\cdot d$	$c\cdot\bar{d}$
$\bar{a}\cdot\bar{b}$				
$\bar{a}\cdot b$	**1**			**1**
$a\cdot b$	**1**			
$a\cdot\bar{b}$	**1**	**1**		

Simplifying SoP and PoS Expressions using Karnaugh Maps

1.

$$f = (a \cdot b \cdot c \cdot d) + (a \cdot b \cdot \bar{c} \cdot d) + (a \cdot \bar{b} \cdot \bar{c} \cdot d) + (a \cdot \bar{b} \cdot c \cdot d) + (\bar{a} \cdot \bar{b} \cdot c \cdot d)$$

$$f = (a + \bar{b} + c + d) \cdot (a + \bar{b} + c + \bar{d}) \cdot (a + b + c + \bar{d}) \cdot (a + b + \bar{c} + \bar{d}) \cdot (\bar{a} + b + \bar{c} + \bar{d})$$

PoS K-map	$c + d$	$c + \bar{d}$	$\bar{c} + \bar{d}$	$\bar{c} + d$
$a + b$		0	0	
$a + \bar{b}$	0	0		
$\bar{a} + \bar{b}$				
$\bar{a} + b$			0	

SoP K-map	$\bar{c} \cdot \bar{d}$	$\bar{c} \cdot d$	$c \cdot d$	$c \cdot \bar{d}$
$\bar{a} \cdot \bar{b}$			1	
$\bar{a} \cdot b$				
$a \cdot b$		1	1	
$a \cdot \bar{b}$		1	1	

$$f = (a \cdot d) + (\bar{b} \cdot c \cdot d)$$

$$f = (a + \bar{b} + c) \cdot (a + b + \bar{d}) \cdot (b + \bar{c} + \bar{d})$$

Class Exercises for Week 12

Even and Odd Parity Checking

Using an **Even Parity Bit Check**, check the following data transmissions by adding your own parity bit in the empty column and ticking the appropriate result:

1	0	0	1	0	0	1	0		Correctly Transmitted ☐ Error Detected ☐
0	1	1	0	0	0	1	0		Correctly Transmitted ☐ Error Detected ☐
0	0	1	1	0	0	1	1		Correctly Transmitted ☐ Error Detected ☐
1	0	0	0	1	1	0	1		Correctly Transmitted ☐ Error Detected ☐

Using an **Odd Parity Bit Check**, check the following data transmissions by adding your own parity bit in the empty column and ticking the appropriate result:

1	1	1	0	0	1	1	0		Correctly Transmitted ☐ Error Detected ☐
0	0	0	1	1	0	1	0		Correctly Transmitted ☐ Error Detected ☐
0	0	1	1	1	1	1	1		Correctly Transmitted ☐ Error Detected ☐
1	0	0	0	0	1	0	1		Correctly Transmitted ☐ Error Detected ☐

Two-Dimensional Even Parity Scheme

Check the transmissions below with a Two-Dimensional Even Parity Scheme by adding your own parity bits to the rows and columns and ticking the appropriate result(s). If you do identify a single bit error reference which bit has the error by entering its Row and Bit Number e.g. R1 B0

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	1	0	1	0	
R2	1	0	0	1	0	0	
R3	0	0	0	1	1	0	
R4	0	1	1	0	0	1	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

2.

	B5	B4	B3	B2	B1	B0	
R1	0	0	1	1	1	1	
R2	0	1	1	1	1	1	
R3	0	1	1	1	1	0	
R4	0	0	1	0	1	0	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

Two-Dimensional Odd Parity Scheme

Check the transmissions below with a Two-Dimensional Odd Parity Scheme by adding your own parity bits to the rows and columns and ticking the appropriate result(s). If you do identify a single bit error reference which bit has the error by entering its Row and Bit Number e.g. R1 B0

1.

	B5	B4	B3	B2	B1	B0	
R1	1	0	1	0	1	0	
R2	1	0	0	0	1	1	
R3	1	1	0	1	1	0	
R4	1	0	0	0	0	0	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

2.

	B5	B4	B3	B2	B1	B0	
R1	1	0	1	1	0	1	
R2	0	1	0	1	0	1	
R3	1	1	0	0	1	0	
R4	1	1	1	1	0	1	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

The Answers for Week 12

Using an **Even Parity Bit Check**

1	0	0	1	0	0	1	0	**1**	Correctly Transmitted ☐ Error Detected ☒
0	1	1	0	0	0	1	0	**1**	Correctly Transmitted ☐ Error Detected ☒
0	0	1	1	0	0	1	1	**0**	Correctly Transmitted ☒ Error Detected ☐
1	0	0	0	1	1	0	1	**0**	Correctly Transmitted ☒ Error Detected ☐

Using an **Odd Parity Bit Check**

1	1	1	0	0	1	1	0	**0**	Correctly Transmitted ☒ Error Detected ☐
0	0	0	1	1	0	1	0	**0**	Correctly Transmitted ☒ Error Detected ☐
0	0	1	1	1	1	1	1	**1**	Correctly Transmitted ☐ Error Detected ☒
1	0	0	0	0	1	0	1	**0**	Correctly Transmitted ☒ Error Detected ☐

Two-Dimensional Even Parity Scheme

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	1	0	1	0	**0**
R2	1	0	0	1	0	0	**0**
R3	0	0	0	1	1	0	**0**
R4	0	1	1	0	0	**1**	**1**
	0	**0**	**0**	**0**	**0**	**1**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R4 B0

2.

	B5	B4	B3	B2	B1	B0	
R1	0	0	1	1	1	1	**0**
R2	0	1	1	**1**	1	1	**1**
R3	0	1	1	1	1	0	**0**
R4	0	0	1	0	1	0	**0**
	0	**0**	**0**	**1**	**0**	**0**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R2 B2

Two-Dimensional Odd Parity Scheme

1.

	B5	B4	B3	B2	B1	B0	
R1	1	0	1	0	1	0	**0**
R2	1	0	0	0	1	1	**0**
R3	**1**	1	0	1	1	0	**1**
R4	1	0	0	0	0	0	**0**
	1	**0**	**0**	**0**	**0**	**0**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R3 B5

2.

	B5	B4	B3	B2	B1	B0	
R1	1	0	**1**	1	0	1	**1**
R2	0	1	0	1	0	1	**0**
R3	1	1	0	0	1	0	**0**
R4	1	1	1	1	0	1	**0**
	0	**0**	**1**	**0**	**0**	**0**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R1 B3

Class Exercises for Week 13

Recap on Even and Odd Parity Checking

Using an **Even Parity Bit Check**, check the following data transmissions by adding your own parity bit in the empty column and ticking the appropriate result:

1	1	0	1	0	1	1	1		Correctly Transmitted ☐ Error Detected ☐
0	1	1	1	0	1	1	0		Correctly Transmitted ☐ Error Detected ☐
1	1	1	1	0	0	1	1		Correctly Transmitted ☐ Error Detected ☐
1	1	0	0	0	1	0	1		Correctly Transmitted ☐ Error Detected ☐

Using an **Odd Parity Bit Check**, check the following data transmissions by adding your own parity bit in the empty column and ticking the appropriate result:

0	0	1	0	0	1	1	0		Correctly Transmitted ☐ Error Detected ☐
1	0	0	0	1	1	1	0		Correctly Transmitted ☐ Error Detected ☐
1	0	1	1	0	1	1	1		Correctly Transmitted ☐ Error Detected ☐
1	1	0	0	0	1	1	1		Correctly Transmitted ☐ Error Detected ☐

Recap on Two-Dimensional Even Parity Scheme

Check the transmissions below with a Two-Dimensional Even Parity Scheme by adding your own parity bits to the rows and columns and ticking the appropriate result(s). If you do identify a single bit error reference which bit has the error by entering its Row and Bit Number e.g. R1 B0

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	0	0	0	0	
R2	0	1	1	0	0	0	
R3	1	0	1	1	1	1	
R4	0	0	0	0	1	1	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: _____

2.

	B5	B4	B3	B2	B1	B0	
R1	1	0	1	0	1	1	
R2	0	1	0	1	1	0	
R3	0	0	1	0	1	0	
R4	0	1	0	1	1	1	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: _____

Recap on Two-Dimensional Odd Parity Scheme

Check the transmissions below with a Two-Dimensional Odd Parity Scheme by adding your own parity bits to the rows and columns and ticking the appropriate result(s). If you do identify a single bit error reference which bit has the error by entering its Row and Bit Number e.g. R1 B0

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	0	0	1	0	
R2	1	0	0	1	1	1	
R3	1	0	0	1	1	0	
R4	0	0	0	1	0	0	

Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: _____

2.

	B5	B4	B3	B2	B1	B0	
R1	0	1	0	1	1	0	
R2	1	0	1	1	1	0	
R3	0	1	0	1	0	1	
R4	0	0	0	0	1	0	

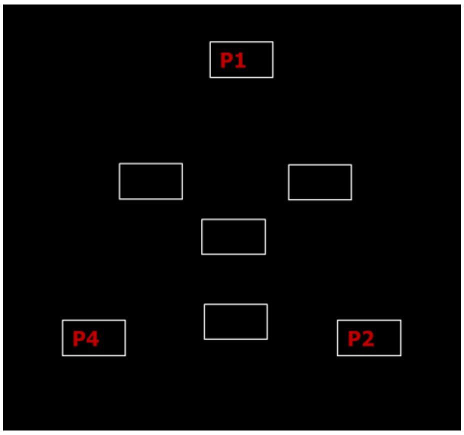
Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: _____

The Hamming Code

Check the transmissions below by completing the Hamming Code diagram and table for each one and then ticking the appropriate result(s). If you do identify a single bit error, reference which bit has the error by entering its bit number e.g. D3

Codeword: 1111000

Diagram

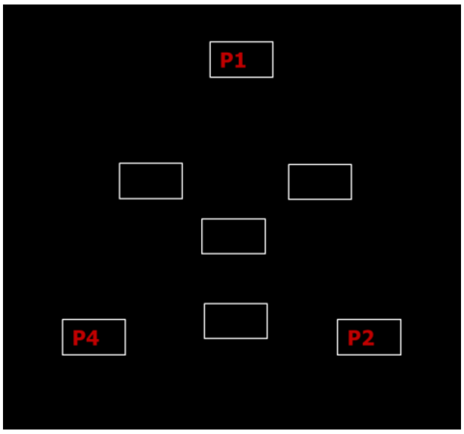


Table

D7	D6	D5	P4	D3	P2	P1	Check

Result:
Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

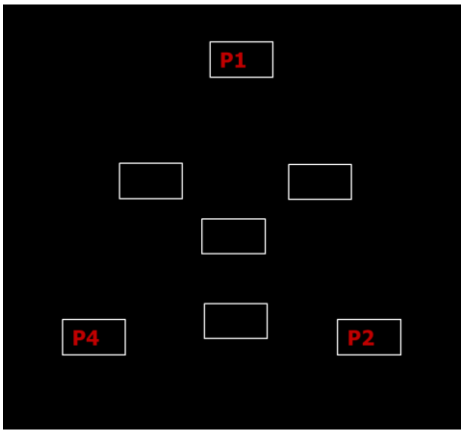
Codeword: 1110001



D7	D6	D5	P4	D3	P2	P1	Check

Result:
Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

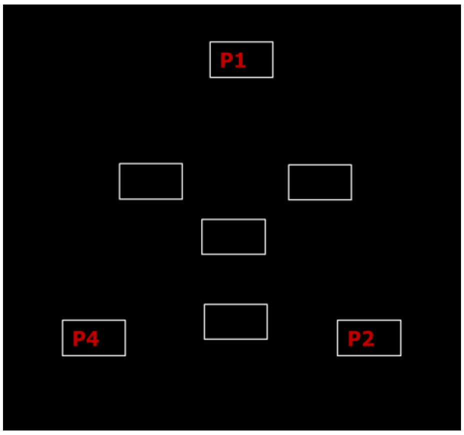
Codeword: 0101001



D7	D6	D5	P4	D3	P2	P1	Check

Result:
Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

Codeword: 0010110



D7	D6	D5	P4	D3	P2	P1	Check

Result:
Correctly transmitted ☐
Error detected ☐
Error can be corrected ☐
Single Bit Error in: ______

Gray Code to Binary Conversion

1. Convert the following gray code to binary and then from binary to decimal:

0010

We have worked this first question through for you.

Gray Code	0	0	1	0
Binary	0	0	1	1

The correct answer is:

$\mathbf{3_{10}}$

2. Convert the following gray code to binary and then from binary to decimal:

0111

Gray Code				
Binary				

The correct answer is:

3. Convert the following gray code to binary and then from binary to decimal:

1101

Gray Code				
Binary				

The correct answer is:

4. Convert the following gray code to binary and then from binary to decimal:

1110

Gray Code				
Binary				

The correct answer is:

5. Convert the following gray code to binary and then from binary to decimal:

1001

Gray Code				
Binary				

The correct answer is:

6. Convert the following gray code to binary and then from binary to decimal:

0110

Gray Code				
Binary				

The correct answer is:

The Answers for Week 13

Using an **Even Parity Bit Check**

1	1	0	1	0	1	1	1	**0**	Correctly Transmitted ☒ Error Detected ☐
0	1	1	1	0	1	1	0	**1**	Correctly Transmitted ☐ Error Detected ☒
1	1	1	1	0	0	1	1	**0**	Correctly Transmitted ☒ Error Detected ☐
1	1	0	0	0	1	0	1	**0**	Correctly Transmitted ☒ Error Detected ☐

Using an **Odd Parity Bit Check**

0	0	1	0	0	1	1	0	**0**	Correctly Transmitted ☒ Error Detected ☐
1	0	0	0	1	1	1	0	**1**	Correctly Transmitted ☐ Error Detected ☒
1	0	1	1	0	1	1	1	**1**	Correctly Transmitted ☐ Error Detected ☒
1	1	0	0	0	1	1	1	**0**	Correctly Transmitted ☒ Error Detected ☐

Two-Dimensional Even Parity Scheme

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	0	0	0	0	**0**
R2	0	1	1	0	0	0	**0**
R3	1	0	1	**1**	1	1	**1**
R4	0	0	0	0	1	1	**0**
	0	**0**	**0**	**1**	**0**	**0**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R3 B2

2.

	B5	B4	B3	B2	B1	B0	
R1	1	0	1	0	1	1	**0**
R2	**0**	1	0	1	1	0	**1**
R3	0	0	1	0	1	0	**0**
R4	0	1	0	1	1	1	**0**
	1	**0**	**0**	**0**	**0**	**0**	**1**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R2 B5

Two-Dimensional Odd Parity Scheme

1.

	B5	B4	B3	B2	B1	B0	
R1	1	1	0	0	1	0	**0**
R2	1	0	**0**	1	1	1	**1**
R3	1	0	0	1	1	0	**0**
R4	0	0	0	1	0	0	**0**
	0	**0**	**1**	**0**	**0**	**0**	**0**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R2 B3

2.

	B5	B4	B3	B2	B1	B0	
R1	0	1	0	1	1	0	**0**
R2	1	0	1	1	1	0	**1**
R3	0	1	0	1	0	1	**0**
R4	0	0	0	0	1	0	**0**
	0	**1**	**0**	**0**	**0**	**0**	**0**

Correctly transmitted ☐
Error detected ☒
Error can be corrected ☒
Single Bit Error in: R2 B4

The Hamming Code

1.

D7	D6	D5	P4	D3	P2	P1	Check
1		1		0		0	**0**
1	1			0	0		**0**
1	1	1	1				**0**

Correctly transmitted ☒

Error detected ☐

Error can be corrected ☐

Single Bit Error in: ______

2.

D7	D6	D5	P4	D3	P2	P1	Check
1		**1**		0		1	**1**
1	1			0	0		**0**
1	1	**1**	0				**1**

Correctly transmitted ☐

Error detected ☒

Error can be corrected ☒

Single Bit Error in: D5

3.

D7	D6	D5	P4	D3	P2	P1	Check
0		0		**0**		1	**1**
0	1			**0**	0		**1**
0	1	0	1				**0**

Correctly transmitted ☐

Error detected ☒

Error can be corrected ☒

Single Bit Error in: D3

4.

D7	D6	D5	P4	D3	P2	P1	Check
0		1		1		0	**0**
0	0			1	1		**0**
0	0	1	**0**				**1**

Correctly transmitted ☐

Error detected ☒

Error can be corrected ☒

Single Bit Error in: P4

Gray Code to Binary Conversion

1. 3_{10}

2. 5_{10}

3. 9_{10}

4. 11_{10}

5. 14_{10}

6. 4_{10}