1. INTRODUCTION

Digital computers have brought about the information age that we live in today. Computers are important tools because they can locate and process enormous amounts of information very quickly and efficiently. They allow us to utilize our mathematical disciplines to the fullest. In one second, a computer can perform calculations that would take a person several days to do by hand. However, computers are not creative and will only do what we tell them. The list of instructions that tells the computer what to do is called a computer program.

System reliability, fast performance, and efficient information storage and retrieval are major factors in the acceptance and use of digital computer systems. The high reliability of computer systems is due largely to the fact that all data is in a digital format. Computers are designed such that digital formatted data can be processed quickly and efficiently.

1.1. Digital Logic States

A computer is made up of many digital circuits that pass information between themselves in the form of digital signals. These signals may represent either instructions or data to be processed. A digital signal can be considered a *logic variable* that has only one of two possible values at any moment in time. These values are called *logic states*. Figure 1.1 describes the common notation for logic states.

Figure 1.1: Logic State Values

TRUE	ON	1	YES
FALSE	OFF	0	NO

The binary (Base 2) number system is often used to represent the states of a group of logic variables at specific instant. Figure 1.2 describes the binary equivalent to the decimal (Base 10) numbers 0 through 9. Only two possible values can exist for each binary digit, either "1" or "0". A binary digit is called a bit and a group of eight bits is called a byte. A thorough discussion of the binary number system, including conversion methods, is discussed in the next section.

Figure 1.2

Base 10	Base 2
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111

2. NUMBER SYSTEMS

The decimal number system (base 10) has become the standard number system used by people for counting and mathematical operations. The decimal (base 10) system is used by most cultures because people have ten fingers. Each finger is used to represent one of ten possible values that a digit can possess.

Digital computers are made up of electronic switches that are considered logic variables with a value of either 1 or 0. When counting in the binary (base 2) number system a bit (digit) may have only a 1 or 0 state (value). A group of eight bits is called a *byte.* A half byte, which is a set of four bits, is often called a *nibble*.

2.1. Binary Numbers

Binary numbers are base two numbers that can be used to represent various integer quantities. The base two number system operates almost the same way as the decimal system; however, only two symbols (0 and 1) exist for each bit while ten symbols (0 through 9) exist for each digit of the decimal system. Both number Systems are right justified. That is, the least significant bit (LSB) of a binary number is always the rightmost bit, just as the least significant digit (LSD) of a decimal number is the rightmost digit.

As described in Figure 2.1, for binary the next significant bit is always a power of two higher than the previous bit. The most significant bit (MSB) represents the highest power of two required for representing a number and is the left-most bit, just as the most significant digit (MSD) of a decimal number is the leftmost digit.

Figure 2.1: Decimal and Binary Number Systems

Humans use Base 10		Computers use Base 2
10 Symbols: 0 to 9		2 Symbols: 0 to 1
72_{10}	=	1001000_2

MSD	LSD		MSB						LSB
7	2_{10}	=	1	0	0	1	0	0	0_2

Powers of 10	Powers of 2
$10^0 = 1$	$2^0 = 1$
$10^1 = 10$	$2^1 = 2$
$10^2 = 100$	$2^2 = 4$
$10^3 = 1000$	$2^3 = 8$

2.1.1. Binary to Decimal Conversion

A binary number is easily converted to decimal notation by summing the powers of two for all bits with a 1 value. Bits with a 0 value are not added to this sum. Therefore, converting the binary number 1011_2 to decimal is performed using the following equation:

$$(1)2^3 + (0)2^2 + (1)2^1 + (1)2^0 = 8 + 2 + 1 = 11_{10}$$

Generally, when working with microcomputers the data is contained in one byte or eight bits modules. Therefore, eight bit conversion will be demonstrated. To convert any binary number to decimal you must determine the power of two corresponding to each bit with a value of 1. Then add up the appropriate magnitudes representing the power of two for each bit. A one byte (eight bit) number is represented by eight bits: $b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0$. The least significant bit, b_0, represents the l's place (2^0), and the most significant bit, b_7, represents the 2^7 or the 128's place. Conversion of an eight bit number is represented by the following equation:

$$b_7 2^7 + b_6 2^6 + b_5 2^5 + b_4 2^4 + b_3 2^3 + b_2 2^2 + b_1 2^1 + b_0 2^0$$

$$b_7(128) + b_6(64) + b_5(32) + b_4(16) + b_3(8) + b_2(4) + b_1(2) + b_0(1)$$

The largest number that can be represented by eight bits would contain all ones for each bit. The binary number $1111{,}1111_2$ equals 255_{10} in decimal. Zero is represented by the binary number $0000{,}0000_2$. Therefore, eight bits may be used to represent any integer decimal number within the range of 0 to 255 inclusive. Additional bits are required to represent decimal integers greater than 255.

Generally, when expressing an eight bit number it is zero filled for all eight bits. For example $1{,}0111_2$ and $0001{,}0111_2$ are equivalent.

2.1.2. Decimal to Binary Conversion

To convert a decimal number to a binary number is more tedious than binary to decimal conversion. A direct mathematical procedure is shown in Figure 2.2. Consider the decimal number 19. Conversion is performed using successive divisions by 2. First 19 is divided by 2 which will generate a quotient of 9 and remainder of 1. Next divide the preceding quotient 9 by 2, which will generate the next quotient of 4 and remainder of 1. Continue the division by 2 until a quotient of 0 exists. The binary equivalent is determined by examining the remainder column and taking the final remainder as the most significant bit and the first remainder as the least significant bit. Therefore, 19 decimal is represented by the binary number 10011. If the binary number is zero filled to eight bits it would be resulted by 0001,0011.

Figure 2.2: Convert 19 to Binary

	Q	R		
19 ÷ 2 =	9	1	(LSB	Least Significant Bit)
9 ÷ 2 =	4	1		
4 ÷ 2 =	2	0		
2 ÷ 2 =	1	0		
1÷ 2 =	0	1	(MSB	Most Significant Bit)

Therefore $19 = 10011_2 = 0001{,}0011_2$

2.1.3. Binary Number Magnitude

The magnitude of decimal numbers which can be represented by binary numbers is restricted by the number of grouped bits. Binary numbers with eight bits can represent unsigned numbers in the range from 255 to 0. Numbers outside of these ranges cannot be represented unless additional bits are used to increase the date word size.

Binary numbers can represent a maximum of 2^n different combinations, where n is the number of bits. Four bits have 16 different possible combinations and can be used to represent decimal numbers in the range 0 to 15. Eight bits have 256 combinations (0 to 255) and sixteen bits have 65,536 combinations (0 to 65,535). For unsigned numbers the range begins at zero.

2.2. Hexadecimal Numbers

Representing eight bits of data as a string of 1's and 0's can be tedious. The hexadecimal number system is used to simplify data representation by encoding 4 bits as one symbol. Hexadecimal numbers comprise the base 16 number system. The hexadecimal number system has sixteen different symbols that represent the value of each hexadecimal digit. As illustrated in the conversion table of Figure 2.3, 0 through 9 are used to represents the first ten hexadecimal symbols, and letters A through F are used to represent the last six symbols. Each hexadecimal symbols represent one of the sixteen possible combinations of four bits. Therefore, eight bits of data is more easily represented in Base 16 by two hexadecimal digits.

Hexadecimal numbers are similar to decimal numbers in that the Most Significant Digit (MSD) is the left most digit and the Least Significant Digit (LSD) is the left most digit. Moving from left to right in order of digits each represents a separate power of sixteen as illustrated below.

Figure 2.3

Base 10	Base 16	Base 2
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

(MSD) (LSD)
A 5 2 F_{16}

(A=)	10	x	4096	=	40,960
(5=)	5	x	256	=	1,280
(2=)	2	x	16	=	32
(F=)	15	x	1	=	15
					$42{,}287_{10}$

2.2.1. Binary to Hexadecimal Conversion

To convert a binary number to hexadecimal is simple. Starting from the least significant bit (b_0), group the binary bits in groups of four. Use the conversion table to determine the hexadecimal symbol that represents each group of four bits.

For example: $1000{,}1100{,}0111{,}1010_2 = 8\ C\ 7\ A_{16}$

2.2.2. Hexadecimal to Binary Conversion

Converting from hexadecimal to binary is just as simple. Use the conversion table to determine the group of four bits, which represents each digit of the hexadecimal number.

For example: $A52F_{16} = 1010,0101,0010,1111_2$

3. ALPHANUMERIC DATA REPRESENTATION

People use written language to communicate among themselves and to give instructions to a computer in the form of a computer program. Alphanumeric data is represented by a binary code for each letter, number, and symbol that is commonly associated with a typewriter keyboard. The ASCII (American Standard Code for Information Interchange) Code is the most commonly used representation for alphanumeric data. The ASCII conversion table of Figure 3.1 is used to convert from either hexadecimal or binary codes to the ASCII characters represented by these codes. All upper and lower case letters, numbers, and symbols used in the English language are in columns 2 through 7. Special computer control characters are found in columns 0 and 1. A definition of each of these control characters is located after the conversion table of Figure 3.1.

An ASCII character is represented by a byte, with bit 7 (MSB) being the parity bit and bits 6 through 0 determined by the conversion table. The parity bit is used for error detection when transmitting data. For the purposes of this class you can assume the parity bit equal to 0.

3.1. Binary String to ASCII Character Conversion

Given a string of binary bits or hexadecimal numbers, conversion to the ASCII characters is performed by separating the string into individual bytes. Assume the parity bit is equal to 0 for this class.

To convert bits 6 through 0 to ASCII, use the ASCII conversion table of Figure 3.2. The least significant nibble, bits 3 through 0, represents the rows of the conversion table. Entries are listed in both binary bits and hexadecimal digits. Bits 6 through 4 represent the columns of the table. Once the specific column and row are found, the ASCII character represented by the byte can be located in the conversion table. Continue this conversion process for each byte of the string. This procedure is illustrated in the example below.

As an example, convert the following hexadecimal representation of a binary string with zero parity to ASCII characters.

Hexadecimal Representation = 576861743F

57	68	61	74	3F	
0101,0111	0110,1000	0110,0001	0111,0100	0011,1111	

Convert using ASCII Conversion Table

W	h	a	t	?	= What?

3.2. ASCII Character to Binary String Conversion

The conversion of a string of characters to a string of bits of hexadecimal digits is performed using the reverse process of above. Find the ASCII character in the ASCII Conversion table. The byte representing this ASCII character is assembled by first determining the row of the character in the table. This specifies the least significant nibble, bits 3 through 0. Then determine the column of the character which specifies bits 6 through 4. Bit 7 is the parity bit and assumed to be 0. Once all eight bits of the ASCII character byte are determined, they can be converted to hexadecimal, if desired. This procedure is illustrated below.

ASCII Character String = What?

	W	h	a	t	?
Conversion	101,0111	110,1000	110,0001	111,0100	011,1111
Parity Added	0101,0111	0110,1000	0110,0001	0111,0100	0011 1111
Hexadecimal	57	68	61	74	3F

Practice Problems

1. Convert the binary numbers to decimal. Verify your answers by converting the decimal result back to binary.

a) 0011 b) 1,1000 c) 1111,1111 d) 1,1100,0000

e) 1011 f) 10,0000 g) 1000,0011 h) 1000,0000,0011

2. Convert the hexadecimal numbers to binary. Verify your answers by converting the binary result back to hexadecimal. All binary numbers should be unsigned 8 bits.

a) 0 b) 9 c) C d) 16

e) 45 f) 63 g) 64 h) AF

3. Write the hexadecimal string for the following ASCII character strings. Assume zero parity. Check your answer by converting the binary string back to an ASCII String.

a) What If?

b) Typhoon!

c) Mangilao, Guam 96923

Figure 3.1: Ascii Conversion Table

HEX	MSD	0	1	2	3	4	5	6	7
LSD	BINARY	000	001	010	011	100	101	110	111
0	0000	NUL	DLE	SPACE	0	@	P	‘	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	{
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	-	=	M	]	m	}
E	1110	SO	RS	.	>	N	^	n	~
F	1111	SI	US	/	?	O	_	o	DEL

Control Characters:

NUL	Null	VT	Vertical Tabulation	SYN	Synchronous Idle
SOH	Start of Heading	FF	Form Feed	ETB	End Transmission Block
STX	Start of Text	CR	Carriage Return	CAN	Cancel
ETX	End of Text	SO	Shift Out	EM	End of Medium
EOT	End of Transmission	SI	Shift In	SUB	Substitute
ENQ	Enquiry	DLE	Data Link Escape	ESC	Escape
ACK	Acknowledge	DC	Device Control 1	FS	File Separator
BEL	Bell	DC	Device Control 2	GS	Group Separator
BS	Backspace	DC	Device Control 3	RS	Record Separator
HT	Horizontal Tabulation	DC	Device Control 4	US	Unit Separator
LF	Line Feed	NA	Negative Acknowledge	DEL	Delete