Binary Multiplication

- **Multiplication is achieved by adding a list of shifted multiplicands according to the digits of the multiplier.**
- **Ex. (unsigned)**

```
   11                    1 0 1 1      multiplicand (4 bits)
X 13                  X   1 1 0 1      multiplier  (4 bits)
-------               ----------------
   33                     1 0 1  1
  11                    0 0 0 0
 _____                 1 0 1 1
  143                  1 0 1 1
                      --------------------
                     1 0 0 0 1 1 1 1      Product (8 bits)
```

Binary Multiplication (continued)

- **Instead of listing all shifted multiplicands and then adding, we can add each shifted multiplicand to a partial product. The previous un-signed example becomes:**

11	1101	multiplicand
x 13	x 1101	multiplier
143	0000	partial product
	1011	shifted multiplicand
	01011	partial product
	0000	shifted multiplicand
	001011	partial product
	1011	shifted multiplicand
	0110111	partial product
	1011	shifted multiplicand
	10001111	product

Two's-complement Multiplication

- A sequence of of two's-complement additions of shifted multiplicands except for last step where the shifted multiplicand corresponding to MSB must be negated.
- Before adding a shifted multiplicand to the partial product, an additional bit is added to the left of the partial product using sign extension.

Ex:

- 5	1011	multiplicand
x - 3	x 1101	multiplier
15	00000	partial product
	11011	shifted multiplicand
Added bit using sign extension →	111011	partial product
	00000	shifted multiplicand
	1111011	partial product
	11011	shifted multiplicand
	11100111	partial product
	00101	shifted and negated multiplicand
	00001111	product

Binary Division

- **Shift and subtract**

Example:

```
      19
11 | 217
     11
    ---
     107
      99
    ---
       8
```

```
          10011   quotient
1011 | 11011001   dividend
       1011       shifted divisor
       ----
        0101      reduced dividend
        0000      shifted divisor
        ----
        1010      reduced dividend
        0000      shifted divisor
        ----
       10100      reduced dividend
        1011      shifted divisor
        ----
       10011      reduced dividend
        1011      shifted divisor
        ----
        1000      remainder
```

Binary Codes

- **Groups of binary bits are often organized in specific ways or binary codes to:**
  - **Represent decimal numbers or alphabetic characters:**
    - **Binary Coded Decimal (BCD).**
    - **American Standard Code for Information Interchange (ASCII)**
  - **Detect errors:**
    - **Even parity code.**
    - **Odd parity code.**
  - **Correct errors:**
    - **CRC Codes.**
  - **Aid in transmission and storage of digital information.**

Binary Coded Decimal (BCD)

- **Binary Coded Decimal (BCD) is a way to store decimal numbers in binary. This number representation uses 4 bits to store each digit from 0 to 9.**

Decimal digit	0	1	2	3	4
BCD	**0000**	**0001**	**0010**	**0011**	**0100**
Decimal digit	5	6	7	8	9
BCD	**0101**	**0110**	**0111**	**1000**	**1001**

- **For example:**

$$1268_{10} = 0001\ 0010\ 0110\ 1000 \quad \text{in BCD}$$

- **BCD wastes storage space since 4 bits are used to store 10 combinations rather than the maximum possible 16.**
- **BCD is often used in business applications and calculators.**

BCD Addition

- **Addition of BCD digits is similar to adding 4-bit unsigned binary numbers except a correction must be made if a result exceeds 1001 by adding 6 to the digit.**

Example:

$$\begin{array}{r} 5 \\ +\ 9 \\ \hline 14 \end{array}$$

$$\begin{array}{rl} & 0101 \\ + & 1001 \\ \hline & 1110 \\ + & 0110 \quad \text{Correction add 6} \\ \hline & 0001\ 0100 \end{array}$$

1 4

Alphanumeric Binary Codes: ASCII

Seven bit codes are used to represent all upper and lower case letters, numbers, punctuation and control characters

	MSBs							
LSBs	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC_1	!	1	A	Q	a	q
0010	STX	DC_2	“	2	B	R	b	r
0011	ETX	DC_3	#	3	C	S	c	s
0100	EOT	DC_4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	O	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Error Detection Binary Codes

- **Errors can occur during data transmission. They should be detected, so that re-transmission can be requested.**
  - **Example: For single-bit error: 0010 can be erroneously transmitted as 0011, or 0000, or 0110, or 1010.**
- **For single-error detection, one additional bit is needed:**
  - **Even parity code:**
  - **additional bit with value to make total number of '1's even.**
  - **Odd parity code: additional bit with value to make total number of '1's odd.**

Character	ASCII Code
0	0110000 1
1	0110001 0
. . .	. . .
9	0111001 1
:	0111010 1
A	1000001 1
B	1000010 1
. . .	. . .
Z	1011010 1
[	1011011 0
\	1011100 1

Odd Parity bits