Number Representation and Arithmetic Circuits

The base of Decimal numbers = 10
any number$v(d) = \sum_{i=0}^{n-1} d_i \times 10^i$
Example : find the value of 895
$v(d) = 5 \times 10^0 + 9 \times 10^1 + 8 \times 10^2$
$= 5 + 90 + 800$

- **Binary Numbers:**
  The base = 2, each digit = 0 or 1
  $v(b) = \sum_{i=0}^{n-1} b_i \times 2^i$

  Example: Find the value of 1101
  $v(b) = 1 \times 2^0 + 0 \times 2^1 + 1 \times 2^2 + 1 \times 2^3$
  $= 1 + 4 + 8 = 13_{10}$

Conversion from decimal to binary

Example: find the binary of 19_{10}
$17 \div 2 = 9$ and remainder 1
$9 \div 2 = 4$ and remainder 1
$4 \div 2 = 2$ and remainder 0
$2 \div 2 = 1$ and remainder 0
$1 \div 2 = 0$ and remainder 1
Binary number is 10011

- **Octal numbers**
  base =8 and digits 0 - 7
  $v(o) = \sum_{i=0}^{n-1} o_i \times 8^i$
  Example: Find the value of 273_8
  $v(o) = 3 \times 8^0 + 7 \times 8^1 + 2 \times 8^2$
  $3 \times 1 + 7 \times 8 + 2 \times 8^2 = 187_{10}$

  Conversion from decimal to octal
  Divide the number by 8
  Example $187 \div 8 = 23$, remainder 3
  $23 \div 8 = 2$, remainder 7
  $2 \div 8 = 0$, remainder 2

Conversion between octal and binary

convert each octal digit to its binary number

Example : $537_8 = (101011111)_2$

Example: $10101 = 010101 = (25)_8$

- **Hexadecimal numbers**

  base =16 and digits 0 - 9, A,B,C,D,E,F

  $v(h) = \sum_{i=0}^{n-1} h_i \times 16^i$

  Example: Find the value of $A2E_{16}$

  $v(o) = 14 \times 16^0 + 2 \times 16^1 + 10 \times 16^2$

  $14 \times 1 + 2 \times 16 + 10 \times 16^2 = 2606_{10}$

  Conversion from decimal to hex

  Divide the number by 16, and consider the remainders

  Conversion between hex and binary

  convert each octal digit to its binary number

  Example : $A2E_{16} = (101000101110)_2$

  Example: $11100101 = 11100101 = (E5)_{16}$

Addition of Unsigned Binary Numbers

Each binary digit is added to the other as:
$0 + 0 = 0$ and carry 0
$0 + 1$ or $1 + 0 = 1$ and carry 0
$1 + 1 = 0$ and carry 1

Example add Y=15 to X= 10
Y= 1111
X= 1010
S= 1001
C= 1110
X+Y= 16+9=25

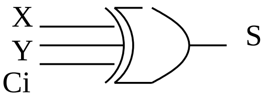
Two bit Adder Circuit Design (Full Adder)

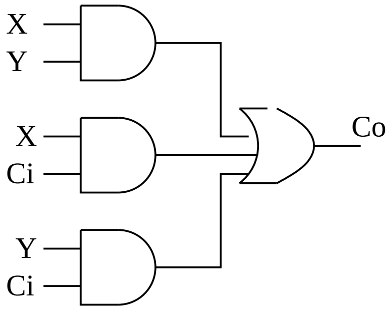
Add X plus Y with Ci, Results = S, and Co

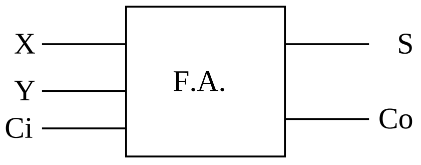
Ci	X	Y	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

S = !Ci.(!X.Y + X.!Y) + Ci.(!X.!Y + X.Y)

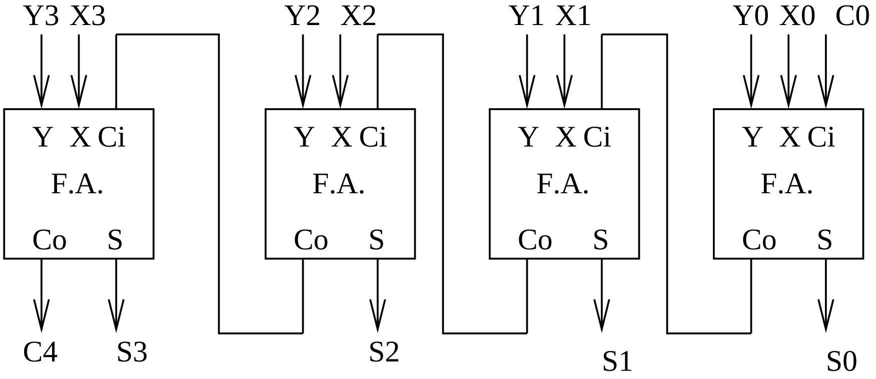
Co = X.Y + Ci.X + Ci.Y







Ripple Carry Adder for N bit Addition



Problem: Propagation Delay of ripple carry

pwdpwd

The Signed Numbers

The Most Significant bit is used for sign.
One less bit to represent the number.
largest number is $= 2^{n-1} - 1$

- **Sign Magnitude:**
  If sign bit (MSB) = 0, the number is positive
  If sign bit =1, the number is negative

  Example: Find the sign magnitude representation of +5 and -5
  +5 the sign bit is 0, +5 = 0101
  -5 the sign bit =1, -5=1101

- **1's Complement:**
  For any number, 1's complement $= (2^n - 1) - k$
  Or invert each bit of the number

  Example: Find 1's complement of +5
  +5= 0101 and -5 in 1's complement is 1010

The 2's Complement

For any number, 2's complement = $(2^n - k)$
1-Invert each bit of the number (1's Complement) then add 1
2- keep digits same until first 1, then invert each following bit

- Example: Find 2's complement representation of +5 and -5
  +5= 0101
  -5 =1010 then add 1 =1011

- Example: Find the decimal value of 2's complement number = 1100
  2's C for 1100 = 0011 + 1= 0100 = 4
  then the number is -4

Addition and Subtraction of signed numbers

It is difficult to add / sub numbers in sign magnitude or 1's complement

Use 2's complement

ADD: use binary addition of the numbers
Subtract: Convert to 2's complement then ADD

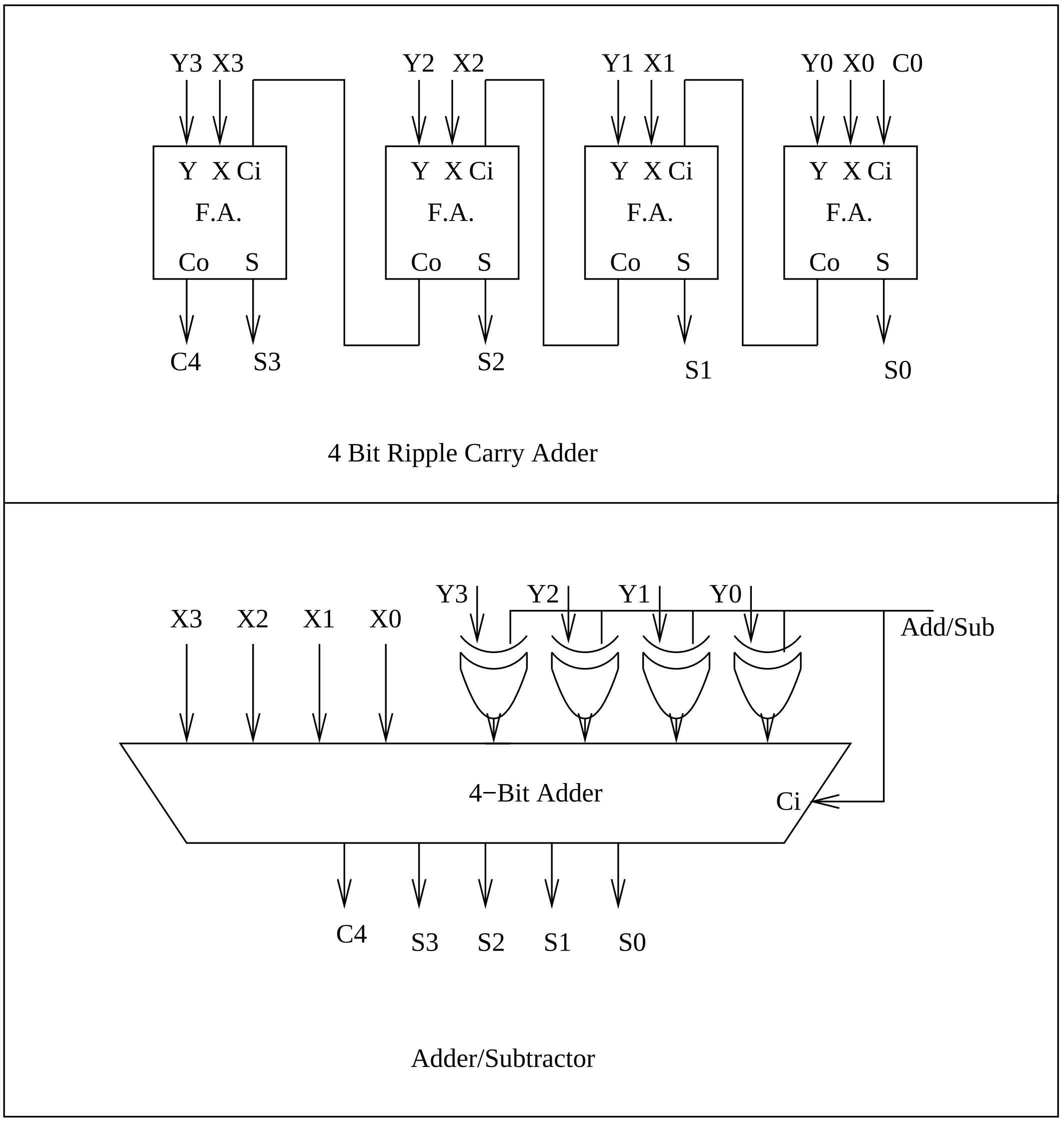
Example: Find +5+2, +5-2, -5+2, -5-2
+5 = 0101, -5= 1011
+2 = 0010, -2= 1110
+5+2=0111,-5-2=1001

-5 = 1011
+2 = 0010
-5+2=1101

Adder/Subtractor

If add/sub=0, operation is X+ Y (add)

If addsub=1, XOR will complement Y and Ci will add 1 and operation is X-Y (sub)



Radix Complement

Can use 10's complemet to decimal numbers:

Example: 74 - 36
10's complement of 36 = (99-36)+1= 64
74-36=74+64=38

Arithmetic Overflow

when the digits are not enough for the result.
Example: using 4 bits to perform +7+6 and -7-6 and -7+6
+7=0111, +6=0110
-7=1001, -6=1010

+7+6=0111+ 0110=1101 a negative result "overflow"
-7-6=1001 + 1010=0011 a positive result "overflow"
-7+6=1001 + 0110=1111 a correct result negative

Overflow Detection

Overflow must be detected in two condidtions:

1- C3=1 and C4=0

2-if C3=0 and C4=1

v=C3.!C4 + !C3.C4= C3 XOR C4

CARRY LOOKAHEAD ADDER (no ripple)

- define generate signal gi=Xi.Yi
- define popagate signal pi=Xi + Yi
- ci+1=gi+ci-1.pi
- c1=g0+c0.p0
- c2=g1+g0.p1+c0.p0.p1
- c3=g2+g1.p2 + g0.p1.p2 +c0.p0.p1.p2
- c4=g3+g2.p3+g1.p2.p3 + g0.p1.p2.p3 + c0.p0.p1.p2.p3
- Si=Xi XOR Yi XOR Ci
- Group generate G0=g7+g6.p7+g5.p6.p7+... c0.p0.p1...p7
  Group propagate P0=p0.p1.p2.p3.p4.p5.p6.p7
  c8=G0 + c0.P0, c16= G1 + G0.P1 + c0.P0.P1

Design of Arithmetic Circuits Using VHDL

VHDL code for full-adder

```
LIBRARY ieee;
USE ieee_std_logic_1164.all;

ENTITY fulladd IS
         PORT(Cin,x,y   :IN  STD_LOGIC;
              s,Cout    :OUT STD_LOGIC);
END fulladd;


ARCHITECTURE LogicFunc OF fulladd IS
BEGIN
  s<= x XOR y XOR Cin;
  Cout<=(x AND y) OR (Cin AND x) OR (Cin AND y);
END LogicFunc;
```

VHDL for 4 bit adder

```
LIBRARY ieee;
USE ieee_std_logic_1164.all;

ENTITY adder4 IS
    PORT(Cin    :IN STD_LOGIC;
         X,Y    :IN STD_LOGIC_VECTOR(3 DOWNTO 0);
         S      :OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
         Cout, OVF :OUT STD_LOGIC);
END adder4;

ARCHITECTURE Behavior OF adder4 IS
     SIGNAL Sum: STD_LOGIC_VECTOR(4 DOWNTO 0);
BEGIN
   Sum<= ("0" & X) +Y + Cin;
   S <= Sum(3 DOWNTO 0);
   Cout <=Sum(4);
   OVF<=Sum(4) XOR X(3) XOR Y(3) XOR Sum(3);
END Behavior;
```

Other Number Representations

- **Floating Point**
  $Value = + - M.2^{E-127}$
  M is mantesa which is LS 23 Bits
  E is exponent and is 8 MS bits
  Sign bit is MSB

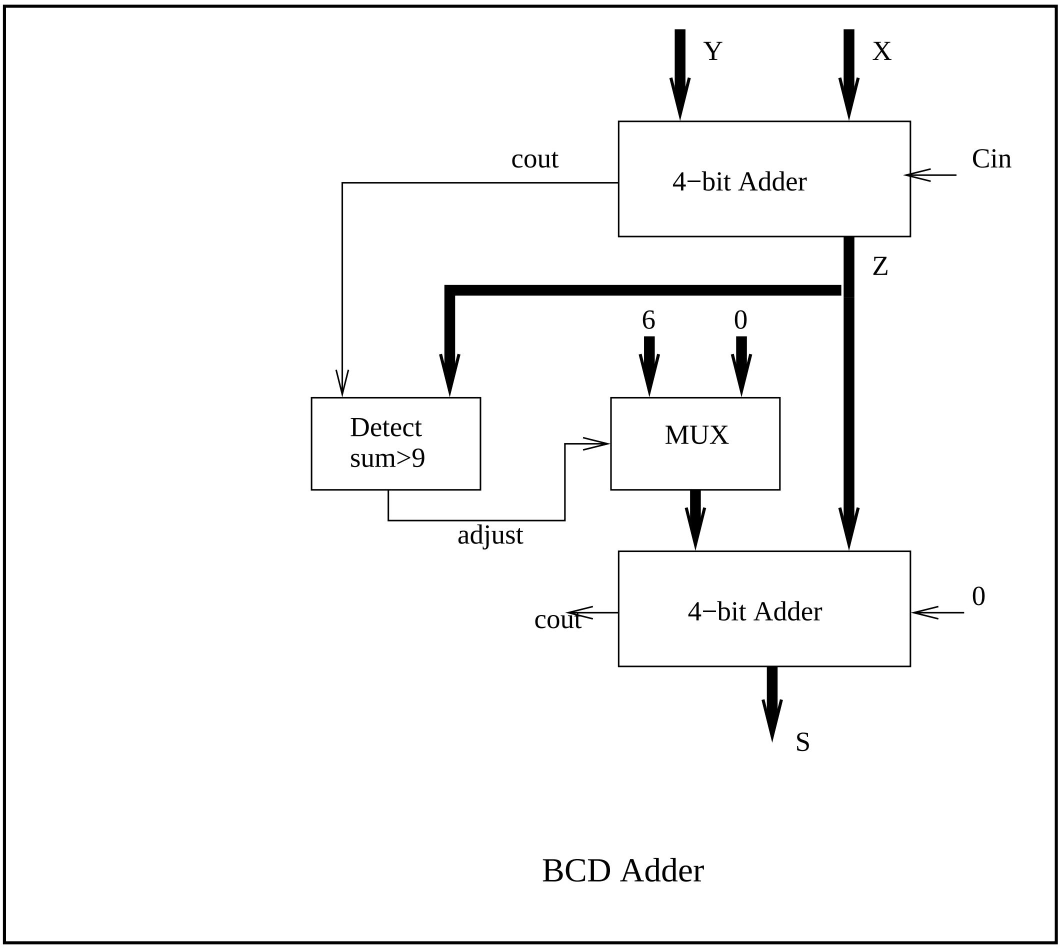
- **BCD "Binary Coded Decimal"**
  It is a code for decimal numbers.
  convert each decimal digit to 4 bit binary

  Example: Find the BCD to 58
  BCD = 0101 1000

- **BCD addition**
  Add each digit in binary, if the result is > 9, then add 6
  Example: 46 + 36
  46= 0100 0110
  36= 0011 0110
  = 0111 1100
  add 6 0110

  = 1000 0010 = 82

BCD Adder



BCD Adder

ASCII Code

Uses 7 bits for 128 characters

number 1 = 0110001 =49

Parity

It uses the 8th bit for parity check.

Even parity: number of 1's = even

Odd parity: number of 1's = odd

Parity Generator and Check For 4 bit generator use XOR

(XOR generates a 1 if number of 1's is odd):

P=X3 XOR X2 XOR X1 XOR X0

At Receiving end C = p XOR X3 XOR X2 XOR X1 XOR X0

If C = 0 no errr, if C=1 then an error occured

Problems from Ch5

5-1-a 0111011110 = 478

b- $(3751)_8 = 1 + 5 \times 8 + 7 \times 64 + 3 \times 8^3 = 2025$

c-$(A25F)_{16} = 15\times 1+5\times 16+2\times 16^2+10\times 16^3 = 41567$

5-3-a 0111011110 = +478

b- 1011100111 = -(0100011001) = -281

5-5 b- +117-34 = 83 no overflow

c- -33 -72 = -105 no overflow

f- -45 + 20 = +25 no overflow

11010011 + 000010100 = 11100111

e-+117 + 42= +159

01110101 + 00101010 = 10011111 over flow

5-19- BCD Subtraction if X= 057, Y =038

Find X-Y

10'S Complement of Y = 999 - 38 +1 = 962

057 + 962 = 019