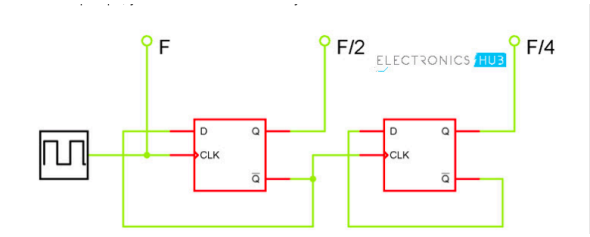
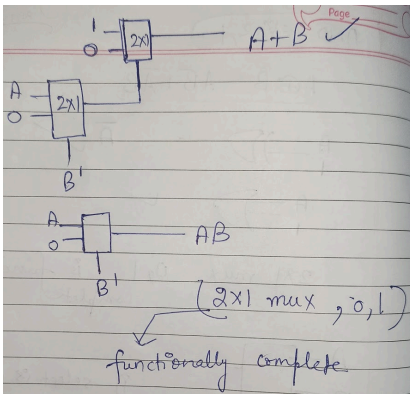
Digital logic design

For a Ring oscillator with Tpd as the propagation delay of Ckts and N is the number of flip flops in the ring oscillator.



For binary counter

$f_{clk} = \frac{1}{2^n}$ where n is the number of flip flops



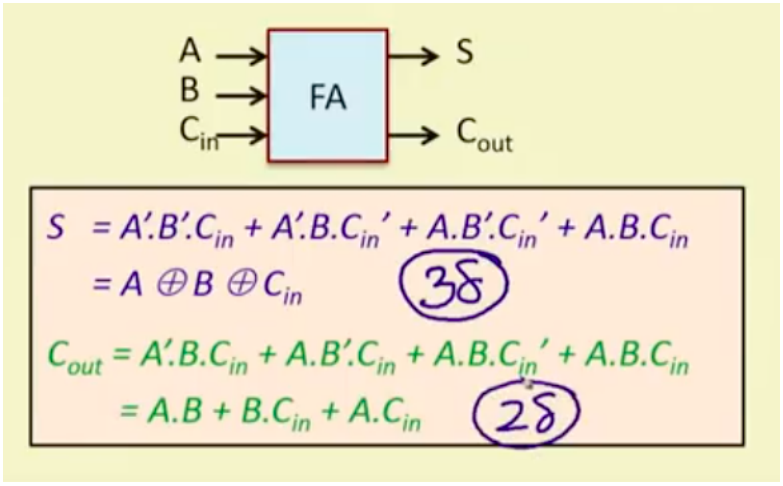
SR - S + R'Q
JK - jQ' + k'Q
T: Q exor T
D = Qn

Consensus Theorem : xy + zy' +xz = zy+zy' as xz is redundant so we can remove it

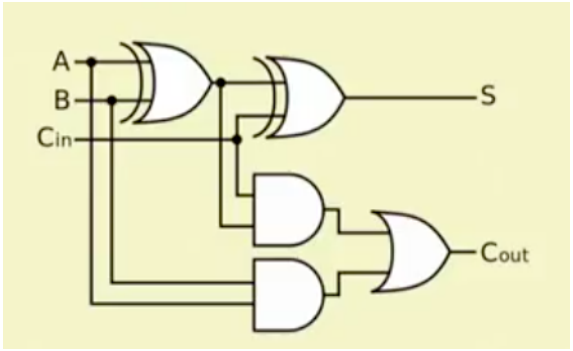
Functional Completeness: using some gates, we can implement any gates

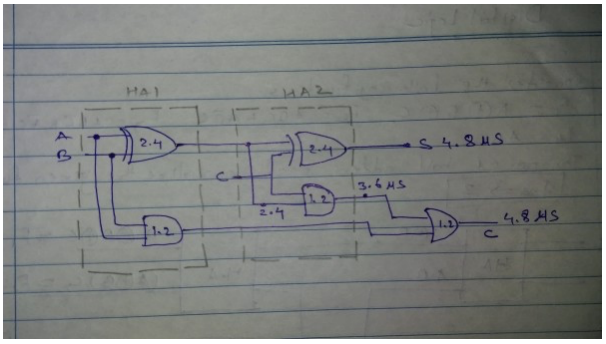
(and, xor, 1)
(and,not)
(or,not)
(2×1 MUX,0,1)
(Or, exor,1)
(And, xor, 1)
All of these are functionally complete

Full Adder

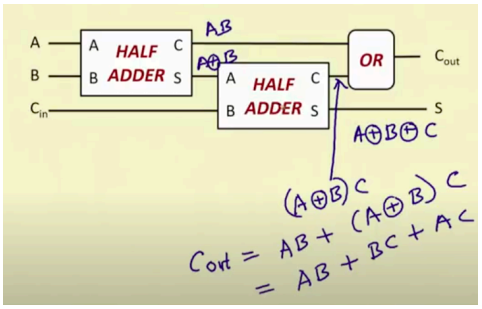


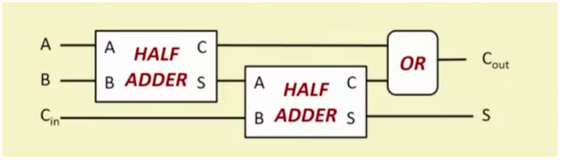
Propagation delay is 3 delta in case of sum as there are three levels namely not, and and OR





Full adder





Ripple carry adder takes 2n+1 gate delays
CLA adder gives 3 + 2 + 3 = 8 time units delay.

If xor gates and its compliments also available then 6 time units delay i.e. 2+2+2 tu

i+1	i	i-1	Recoded 1-bit pair (i+1 i)	2-bit Booth
0	0	0	(0 0)	0
0	0	1	(0 1)	1
0	1	0	(1 -1)	1
0	1	1	(1 0)	2
1	0	0	(-1 0)	-2
1	0	1	(-1 1)	-1
1	1	0	(0 -1)	-1
1	1	1	(0 0)	0

Booth's Algorithm
Append 0 at rhs
Then 0--->1 transition is +1 and 1---->0 transition is -1 rest is 0

Johnson counter: 2N states
Ring counter : N states

N is the number of flip flops used in the implementation of the counter

In conclusion, the value *x* represented by the word can be determined based on the following rules, including all the exceptional cases:

- If *E* = 255 and *F* is nonzero, then *x* = *NaN* ("Not a number").
- If *E* = 255, *F* is zero, and *S* is 1, then *x* = – Infinity.
- If *E* = 255, *F* is zero, and *S* is 0, then *x* = + Infinity.
- If 0 < *E* < 255, then $x = (-1)^s \times (1.F) \times 2^{E-127}$, where 1. *F* represents the binary number created by prefixing *F* with an implicit leading 1 and a binary point.
- If *E* = 0 and *F* is nonzero, then $x = (-1)^s \times (0.F) \times 2^{-126}$. This is an "unnormalized" value.
- If *E* = 0, *F* is zero, and *S* is 1, then *x* = –0.
- If *E* = 0, *F* is zero, and *S* is 0, then *x* = 0.

Typical and exceptional examples are shown as follows:

$0\ 00000000\ 00000000000000000000000 = 0$
$1\ 00000000\ 00000000000000000000000 = -0$
$0\ 11111111\ 00000000000000000000000 = \text{Infinity}$
$1\ 11111111\ 00000000000000000000000 = -\text{Infinity}$
$0\ 11111111\ 00000100000000000000000 = \text{NaN}$
$1\ 11111111\ 00100010001001010101010 = \text{NaN}$
$0\ 00000001\ 00000000000000000000000 = (-1)^0 \times (1.0_2) \times 2^{1-}$
$0\ 00000000\ 10000000000000000000000 = (-1)^0 \times (0.1_2) \times 2^{0-}$
$0\ 00000000\ 00000000000000000000001 = (-1)^0 \times (0.0000000$
$= 2^{-149}\ (\text{smallest positive value})$

For normalized value in IEEE-754 representation the exponent field cannot be all $1s$. So, to get the maximum exponent we should make the exponent field 1111110 which equals 254 but with a bias of 127 (used to have negative exponent in IEEE-754 representation), this equals 127.

Sign bit must be 0 to make the number positive.

Mantissa bits must be all $1s$ to maximize the number so that the represented number equals $1.\underbrace{111..1}_{23\ 1s} = 1 + (1 - 2^{-23}) = 2 - 2^{-23}$.

So, correct option is D and the represented value $= (2 - 2^{-23}) \times 2^{127}$

A logic circuit implements the following Boolean function: $F = A'C + ACD'$ It is found that the circuit input combination $A = D = 1$ can never occur. The simplified expression for F using the proper don't-care conditions is

A. $F = A'C$
B. $F = CD'$
C. $F = C$
D. $F = A'C + CD'$

F=A'C + ACD' = C (A' + AD') = C (A'+D')

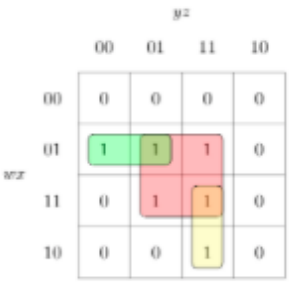
A decimal number has 32 digits. The minimum number of digits required to accurately represent the number in binary is _____.

9999...999 32 times is largest number,
Can be written as

$10^{32} - 1 \leq 2^n - 1$ $\qquad 10^{32} - 1$
$n = ceil\left(32 \times \log_2 10\right)$ =107

Given that A=D=1 never occur, SO we never encounter D'=0 and A'=0 simultaneously. That means either A' = 1 or D'=1.

F = C (A'+D') = C (1) = C



A.

We avoid a static-1 hazard if every adjacent $1s$ in K-map is covered by an implicant. In Option A, this is TRUE. In all other options, at least 2 adjacent $1s$ are there which are not being covered by an implicant.

http://www.ee.scu.edu/classes/2000fall/elen021/supp/stathaz.html

Let A be the maximum value represented in IEEE-754 single-precision representation and B be the maximum value represented in IEEE-754 double precision representation (both ignoring infinities and other special values). B/A approximates to

A. 10^{23}
B. 10^{270}
C. 10^{99}
D. 10^{308}

IEEE-754 single precision representation: $\underbrace{1 \text{ bit}}_{\text{Sign}}\ \underbrace{8 \text{ bits}}_{\text{Exponent}}\ \underbrace{23 \text{ bits}}_{\text{Mantissa}}$

Maximum value, $A = (2 - 2^{-23}) \times 2^{127}$

IEEE-754 double precision representation: $\underbrace{1 \text{ bit}}_{\text{Sign}}\ \underbrace{11 \text{ bits}}_{\text{Exponent}}\ \underbrace{52 \text{ bits}}_{\text{Mantissa}}$

Maximum value, $B = (2 - 2^{-52}) \times 2^{1023}$

$$B/A = \frac{2 - 2^{-52}}{2 - 2^{-23}} \times 2^{896} \approx 10^{896 \times \log 2} \approx 10^{270}$$

Denormalized numbers are used to extend the range of numbers between $0 - 1$ both on the positive as well as negative side that can be represented by normalized numbers. i.e., they can represent smaller positive numbers and larger negative numbers than that can be represented by normalized numbers in IEEE-754 representation.

Exponent Value	Mantissa	Represents
11111111	All zeros	Infinity (∞)
11111111	Not all zeros	Not a number (NAN)
00000000	All zeros	Zero
00000000	Not all zeros	Subnormal (very small)