N: Positive Integer
Z: Integer

Comp 2012

to form logic expression

Logic and Proofs 逻辑与命题

用 truth table 证明逻辑等价

↖ Logic Operator

一、Proposition Logic (全都可用 truth table 证明)

1、negation(not): $\neg$ conjunction(and): $\wedge$ disjunction(or): $\vee$

2、Implication: $p\to q \equiv \neg p \vee q$ if ··· then ··· premise(前提) 如果p成立，q必成立

3、Contrapositive(逆否): $\neg q \to \neg p \equiv p \to q$

4、Biconditional (双条件): $p \leftrightarrow q \equiv (p\to q)\wedge(q\to p)$ ⇒ 异或

p	q	p↔q
T	T	T
T	F	F
F	T	F
F	F	T

5、Tautology (always true): $p \vee \neg p \equiv T$

6、Contradiction (always false): $p \wedge \neg q \equiv F$
↳ Logically Equivalence

De Morgan's Laws: $\neg(p\vee q) \equiv \neg p \wedge \neg q$
(可用truth table证明) $\neg(p\wedge q) \equiv \neg p \vee \neg q$

二、Propositional Function and Quantifier

(一) like, Let $P(x,y)$ denote the statement "x is less then y"

Truth Table

x	y	P(x,y)
1	1	F
1	2	T
2	1	F
2	2	F
⋮	⋮	

Condensed truth table ⇒ 可以表示多元情况

	x=1	x=2	···	x=n
y=1	T/F	T/F		T/F
y=2	T/F			
⋮				
y=n	T/F			

对于 $A\cup B$: $x\in A \vee x \in B$ $A\cap B$: $x\in A\wedge x\in B$ $A-B$: $x\in A$ 且 $x\notin B$

证明 $A\subseteq B$: $\forall x\{x\in A\to x\in B\}$ $A\supseteq B$: $\forall x\{x\in B\to x\in A\}$

$A=B$: $\forall x\{x\in A\leftrightarrow x\in B\}$

(二) Universal quantifier $\forall$: $\forall x\in D\ P(x)$; $P(x)=T \equiv P(x_1)\wedge P(x_2)\cdots\wedge P(x_n)=T$

Existential quantifier $\exists$: $\exists x\in D\ P(x)$; $P(x)=T \equiv P(x_1)\vee\cdots\vee P(x_n)=T$

De Morgan's Laws: $\neg\forall x\in D\ P(x) \equiv \exists x\in D\ \neg P(x)$
$\neg\exists x\in D\ P(x) \equiv \forall x\in D\ \neg P(x)$

Nested Quantifier (打印P35)

三、Rules of Interface
divide them into several steps, and deduce statement
ex: P30 P31

四、Proof methods

1. direct proof: $p\to q$ 2. proof by contraposition (逆否命题) $p\to q\equiv\neg q\to\neg p$ P33
3. contradiction $\neg[\forall x\in N\ P\to q] \equiv \exists x\in N\ p\wedge\neg q$ ∃变∀，条件不否，否结论
4. prove by case 分类讨论 5、对于存在性证明：可举例，反证法
6. proof by induction 数学归纳法

Rules of inference

- Given the following premises
  - $\neg p \wedge q$
  - $r \to p$
  - $\neg r \to s$
  - $s \to t$
- Show that the statement t is true

	Premises		Deduce a statement
Step 1: apply rule 8	$\neg p \wedge q$		$\neg p$
Step 2: apply rule 2	$\neg p$ (from step 1)	$r \to p$	$\neg r$
Step 3: apply rule 1	$\neg r$ (from step 2)	$\neg r \to s$	s
Step 4: apply rule 1	s (from step 3)	$s \to t$	t

Nested Quantifiers

- Assume that the domain $D=\{1,2,3\}$
- For each statement below, identify the matching table (with the fewest "T" entries)
  1. $\forall x\in D\ \forall y\in D\ P(x,y)$
  2. $\forall x\in D\ \exists y\in D\ P(x,y)$
  3. $\exists x\in D\ \forall y\in D\ P(x,y)$
  4. $\exists x\in D\ \exists y\in D\ P(x,y)$
  5. $\forall y\in D\ \forall x\in D\ P(x,y)$

2	x=1	x=2	x=3
y=1	F	F	T
y=2	T	F	F
y=3	F	T	F

1/5	x=1	x=2	x=3
y=1	T	T	T
y=2	T	T	T
y=3	T	T	T

4	x=1	x=2	x=3
y=1	F	F	F
y=2	F	F	F
y=3	F	T	F

3	x=1	x=2	x=3
y=1	F	T	F
y=2	F	T	F
y=3	F	T	F

24

重写 function + 做tut的题

Algorithm

一、Syntax Convention P12

二、$O(n)$: $\exists$ positive c, n_0, let $0\le f(n)\le cg(n)$ for all $n\ge n_0$

$\Omega(n)$: $0\le g(n)\le cf(n)$

$\Theta(n)$: $\exists$ positive c_1,c_2,n_0, let $0\le c_1g(n)\le f(n)\le c_2g(n)$ for all $n\ge n_0$

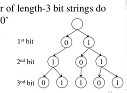
ex: $T(n)$ is $O(n)$: $0\le T(n)\le cn$
$T(n)$ is $\Omega(n)$: $0\le cn\le T(n)$
$T(n)$ is $\Theta(n)$: $0\le c_1n\le T(n)\le c_2n$

$O(n)$ is $O(n^2)$ ✓ correct but not accurate

$\underbrace{O(n)+O(n)+\cdots+O(n)}_{n\ terms} = (c_1+\cdots+c_n)n = (cn)n = O(n^2)$

Tree diagram

- Tree diagram may be used to solve a counting problem
  - It starts with the root
  - Each branch indicates a possible choice
    - Only keep branches that satisfy constraints in the problem!
- *Example*: count the number of length-3 bit strings do not have two consecutive '0'
  - Each bit can be either 0 or 1
  - The next bit cannot be 0 if the previous bit is 0
  - The answer = 5



13

Basic Structure

一、Set ⇒ $x\in S$ logic expression

1、P9 ⇒ 证明集合关系 (midterm)

2、$\bigcup_{i=1\cdots n} S_i \equiv S_1\cup S_2\cup\cdots\cup S_n$ $\bigcap_{i=1\cdots n} S_i = S_1\cap S_2\cap\cdots\cap S_n$

3、Cartesian Product $S\times T=\{(s,t)\mid s\in S\wedge t\in T\}$
$S=\{a,b\}$ $T=\{c\}$ $S\times T=\{(a,c),(b,c)\}$

4、Venn Diagram ⇒

5、Set Identities ①Exchange ✓
②Combination $S\cup(T\cap R)=(S\cup T)\cap(S\cup R)$

二、Sequences

三、Matrices

四、Function

1、one-on-one 单射：1 VS 1

onto 满射：for all y, each y has at least 1 x

五、Relation → Binary Relation: subset of set X × set Y

Equivalence relation:
- Reflexive: $\forall x\in X$, $(x,x)\in R$
- Symmetric: $\forall x,y\in X$, $(x,y)\in R\to(y,x)\in R$
- Transitive: $\forall x,y,z\in X$, $(x,y)\in R\wedge(y,z)\in R \to (x,z)\in R$

antisymmetric: $\forall x,y\in R$, $(x,y)\in R\wedge(y,x)\in R\to x=y$
↳ partial ordering: Hasse diagram 用于 partial ordering 可视化
↳ Reflection, Antisymmetric, Transitive

$S_n = \frac{a_1(1-q^n)}{1-q} = \frac{a_1(q^n-1)}{q-1}$

✻ midterm 第4题 set-builder notation

Even $=\{x\in N\mid (x\le 8)\wedge(\exists k\in N(x=2k))\}$

$T=\{x\in N\mid(\exists a\in N(x=5a))\wedge(\forall b\in N(x\neq 7b))\}$

✻ sequence ↑: $\forall i\ a_i\le a_{i+1}$ ↓: $\forall i\ a_i\ge a_{i+1}$

subsequence $a=(1,2,4,8,16)$ $c=(1,4,8)$ c is subsequence of a

ex: $\sum_{i=1\cdots3}\sum_{j=1\cdots2}(i*j)$
$=\sum_{i=1\cdots3}(i+2i)=\sum_{i=1\cdots3}(3i)=3+6+9=18$

✻五、Equivalence class

等价关系可将集合X划分(partition)为互不相关的等价类，

所有等价类的并集就是原集合X

ex: $[x]_R=\{(x,y)\in N\times N\mid (x-y) \bmod 3=0\}$ ⟺ ①+②+③ 等价关系

① $[1]_{R\,MOD_3}=\{1,4,7,\cdots\}$
② $[2]_{R\,MOD_3}=\{2,5,8,\cdots\}$
③ $[3]_{R\,MOD_3}=\{3,6,9,\cdots\}$
} 等价类

✻ prove $\underbrace{A-(A-B)}_{\text{转化成 Logic expression}}=A\cap B$

$A\cap\neg(A-B)$	$x\in A\wedge x\notin(A-B)$
$A\cap\neg(A\cap\neg B)$	$x\in A\wedge\neg(x\in A\wedge x\notin B)$
$A\cap(\neg A\cup B)$	$x\in A\wedge(x\notin A\vee x\in B)$
$(A\cap\neg A)\cup(A\cap B)$	$(x\in A\wedge x\notin A)\vee(x\in A\wedge x\in B)$
$A\cap B$	false $\vee(x\in A\wedge x\in B)$
	$x\in A\wedge x\in B$

Counting

一、Basic P

$|S_1\cup S_2\cup\cdots\cup S_n|=\sum_{k=1}^{n}|S_k|-\sum_{k=1}^{n-1}\sum_{m=k}^{n}|S_k\cap S_m|+\cdots-(-1)^{n-1}\left(\sum_{k=1}^{n-t+1}\cdots\sum_{z=1}^{n}\right)|S_k\cap\cdots\cap S_t|$

二、The pigeonhole principle 鸽巢原理 ①平均分布 ②分布中的某一个，而非具体哪个

generalized pigeonhole principle P21, 22, 23

ex: throw k balls into t boxxes, one box has at least $\lceil \frac{k}{t}\rceil$ balls

{扑克牌中的一种花色
{扑克牌中的拣心花色

$P(n,r)=\frac{n!}{(n-r)!}$ $C(n,r)=\frac{n!}{r!(n-r)!}$

三、Generalized permutations and combinations

✻ $(x+y)^n=\sum_{r=0}^{n}\binom{n}{r}x^{n-r}y^r$ $2^n=\sum_{r=0}^{n}\binom{n}{r}$ ✻ $\binom{n+1}{r}=\binom{n}{r-1}+\binom{n}{r}$

1、Combinations with repetitions
throw n balls into t kinds of boxes,
there are $C(n+r,n-1)$ combinations.

2、Permutations with indistinguishable objects
ex: hahaha 这6个字母可生成多少种单词
result $=\frac{n!}{n_1!n_2!\cdots n_n!}=\frac{6!}{3!3!}$

✻二、① 有n个隔子，但有n+1个物品，必有1个重复
② 使用2个序列，1个不变，另外1个用加法，2个序列合并，复用①

✻ 52张扑克牌取4张的总可能数 $C_{52}^4 \neq \cancel{C_{52}^1C_{51}^1C_{50}^1C_{49}^1}$

Set Identities

- How to use basic definitions to prove an identity?
- How to use an identity?

Identity
$S\cup\varnothing=S$
$S\cap\varnothing=\varnothing$
$S\cup T=T\cup S$
$S\cap T=T\cap S$
$(S\cup T)\cup R=S\cup(T\cup R)$
$(S\cap T)\cap R=S\cap(T\cap R)$
$S\cup(T\cap R)=(S\cup T)\cap(S\cup R)$
$S\cap(T\cup R)=(S\cap T)\cup(S\cap R)$

X={1,2,3,4}

R = {(2,2), (2,3), (2,4), (3,2), (3,3), (3,4)}

whether R is **transitive**.

Let's check all the combinations:

	(2,2)	(2,3)	(2,4)	(3,2)	(3,3)	(3,4)
(2,2)	(2,2)	(2,3)	(2,4)	/	/	/
(2,3)	/	/	/	(2,2)	(2,3)	(2,4)
(2,4)	/	/	/	/	/	/
(3,2)	(3,2)	(3,3)	(3,4)	/	/	/
(3,3)	/	/	/	(3,2)	(3,3)	(3,4)
(3,4)	/	/	/	/	/	/

Description	Notation	Result
Subset	$S\subseteq T$	$\forall x\,(x\in S\to x\in T)$
Superset	$S\supseteq T$	$\forall x\,(x\in T\to x\in S)$
Set equality	$S=T$	$\forall x\,(x\in S\leftrightarrow x\in T)$
Union	$S\cup T$	$\{\,x\mid x\in S\vee x\in T\,\}$
Intersection	$S\cap T$	$\{\,x\mid x\in S\wedge x\in T\,\}$
Difference	$S-T$	$\{\,x\mid x\in S\wedge x\notin T\,\}$

Syntax Convention (used in our slides)

Their syntaxes have minor differences!

	Our algorithms	*Programming language (Java/C)*
Block structure	By indentation level	By explicit begin/end like '{' '}'
Assignment operator	←	=
Equality test operator	=	==
Array index	Start from 1	Start from 0
Use of English words	for primitive steps	No

Selection-Sort (Array A, Integer n)

```
1. for integer i ← 1 to n–1
2.     k ← i
3.     for integer j ← i+1 to n
4.         if  A[k] > A[j]  then
5.         ……
```

Algorithm

void Selection-Sort (int* A, int n)

```
int i, j, k, temp ;
for ( i=0 ; i<n–1; i++ ) {
    k = i ;
    for ( j=i+1 ; j<n; j++ )
        if  ( A[k] > A[j] )
        ……
}
```

C program

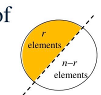
11

Combinatorial proof

- What is a combinatorial proof of an identity?
  - (i) *double counting proof*:
    count the number of objects in two different ways,
    OR
  - (ii) *bijective proof*:
    show the one-to-one correspondence between two sets of objects
- E.g., a combinatorial proof of C(n,r) = C(n,n–r)

28

Combinatorial proof



- Combinatorial proof of C(n,r) = C(n,n–r)
  - Consider a double counting proof
  - Let S be a set of n elements
  - The first way to count the number of size-r subsets of S
    - By definition, C(n,r) is the number of size-r subsets of S
  - The second way to count the number of size-r subsets of S
    - For each size-(n–r) subset A of S, the corresponding S–A is a size-r subset of S
    - Thus, we have C(n,n–r) size-r subsets of S
  - Therefore, C(n,r) = C(n,n–r)

做一下 MI example

Induction and Recursion

一、Mathematical Induction

Basic Step: $P(1) = T$

Inductive Step: try to show $P(k) \to P(k+1)$

may have mistake

so

二、Strong Induction

Basic Step: $P(1) = T$

Inductive Step: $P(1) \cap P(2) \cap \cdots \cap P(k) \to P(k+1)$

Let $P(n)$ be the propostion that n can be expressed as the product of prime numbers

Basic step:

2 is prime, $P(2)$ is true

Inductive step:

assume $P(k)$ is true for all j in $[2,k]$,

so ① if $(k+1)$ is prime

$P(k+1)$ is true

② if $(k+1)$ is not prime

$\exists\, a \cdot b\ (2 \le a \le b \le k)$, let $k+1 = a \cdot b$

$\because P(a), P(b)$ is true

$\therefore P(k+1)$ can be expressed above

$\therefore P(k+1)$ is true

三、Recursive algorithms

1. Tower of Hanoi $T(n) = 2^n - 1$

```
SovleHanoi(integer n, rod A, rod B, rod C)
  if n ≥ 0
    SovleHanoi(n-1, S, R, T)
    MoveHanoi(S, T)
    SovleHanoi(n-1, R, T, S)
```

2. MergeSort(Array A[1⋯n])

```
if n > 1
  m ← n/2
  B[1⋯m] ← A[1,⋯m]
  C[1⋯n-m] ← A[m+1,⋯n]
```

```
Merge-Sort(B[1⋯m])
Merge-Sort(C[1⋯n-m])
A[1⋯n] ← Merge(B[1⋯m], C[1⋯n-m])

Merge(Array L[1⋯nL], Array R[1⋯nR])
nA ← nL + nR
create new array A[1⋯nA]
i ← 1, j ← 1
for k ← 1 to nA
  if (i ≤ nL) and (j ≥ nR or L[i] ≤ R[j])
    A[k] ← A[i]; i ← i+1;
  else
    A[k] ← A[j]; j ← j+1;
return A;
```

Binomial coefficients

- Consider the following recursively definition of $C(n, k)$
  - *Basis step*

    $C(n, 0) = C(n, n) = 1$ for any $n > 0$
  - *Recursive step*

    $C(n, k) = C(n-1, k-1) + C(n-1, k)$, for $0 < k < n$
- In fact, $C(n,k)$ is a binomial coefficient
  - The number of size-k combinations of n elements
- How to evaluate $C(3,1)$?
  - Find it manually, or
  - Run an algorithm

24

A recursive algorithm for finding $C(n,k)$

Basis step

$C(n, 0) = C(n, n) = 1$ for any $n > 0$

Recursive step

$C(n, k) = C(n-1, k-1) + C(n-1, k)$, for $0 < k < n$

C (n, k)

1. if $k = 0$ or $k = n$
2. return 1
3. else
4. $a \leftarrow C(n-1, k-1)$
5. $b \leftarrow C(n-1, k)$
6. return $(a + b)$

However, this algorithm is extremely slow when n is large.

You will learn how to design faster algorithms in COMP3011.

26

Q2

Question 2

Let $\Psi = \{‘0’, ‘1’\}$. **Write** a recursive definition of the set of binary strings that have more zeros than ones.

The set of the above binary strings Ψ^* is defined as:

// basis step

$0 \in \Psi^*$

// recursive step

$x \in \Psi^* \wedge y \in \Psi^* \wedge z \in \Psi \to$ $xy \in \Psi^*$, $xyz \in \Psi^*$, $xzy \in \Psi^*$, $zxy \in \Psi^*$

Notice 01 is not eligible

24

Q3

Question 3

Write a recursive algorithm to compute $x^n \bmod m$, where x, n, m are positive integers.

[Hint: use the identity $(a*b) \bmod m = ((a \bmod m)*(b \bmod m)) \bmod m$]

PowerMod(unsigned int x, unsigned int n, unsigned int m):

1. if n = 1:
2. return x mod m
3. B ← PowerMod(x, ⌊ n/2 ⌋ , m)
4. $A \leftarrow (B * B) \bmod m$
5. if n is odd:
6. A ← (A * x) mod m
7. Return A

$x^n = x^{\frac{n}{2}} * x^{\frac{n}{2}}$ if n is even,

$x^n = x^{\lfloor \frac{n}{2} \rfloor} * x^{\lfloor \frac{n}{2} \rfloor} * x$ if n is odd

The sets of all strings

- We are given a set of characters Ψ, called alphabet
- Let ε denote the empty string
- The set Ψ^* of all strings over the alphabet is defined as:
  - *Basis step*

    $\varepsilon \in \Psi^*$
  - *Recursive step*

    $x \in \Psi^* \wedge y \in \Psi \to xy \in \Psi^*$

    where xy means the string formed by placing y at the end of x

26

Merge Sort

```
Merge-Sort ( Array A[1..n] )
1. if n > 1
2.   m ← ⌊ n/2 ⌋
3.   B[1.. m] ← A[1..m]
4.   C[1..n–m] ← A[m+1..n]
5.   Merge-Sort ( B[1..m] )
6.   Merge-Sort ( C[1..n–m] )
7.   A[1..n] ← Merge (B[1.. m], C[1..n–m] )
```

Divide the array into two sub-arrays of n/2 numbers each

Sort two sub-arrays recursively

Merge two sorted sub-arrays into a sorted array

Boolean Operator:

$A \cup B$: $0+0=0$, $0+1=1$, $1+0=1$, $1+1=1$

$A \cap B$: $0 \cdot 0 = 0$, $0 \cdot 1 = 0$, $1 \cdot 0 = 0$, $1 \cdot 1 = 1$

Boolean Expression: E, $E_1 + E_2$, $E_1 \cdot E_2$ ⇒ Boolean Function 2^{2^n}

Boolean Algebra & Circuits

一、Basic

1. Known Identities 特性 $P_{11,12}$ prove an Identities 证明2个式子相等，用truth table
2. Duality 对偶性 $+ \Leftrightarrow \cdot$　$0 \Leftrightarrow 1$ ← 2种证明方法 + 逻辑化简

只需证明一个恒等式正确，就可得到另一个恒等式正确

ex: 证 $x + y \cdot z = (x+y) \cdot (x+z)$ ✓　$x+y$ 的对偶式为 $x \cdot y$

↳ 即证: $x \cdot (y+z) = xy + xz$ ✓

3. 异或的表示: $F(x,y) = \bar{x} \cdot \bar{y} + x \cdot y$

二、Logic Gate

① NOT Gate　③ OR Gate　④ AND Gate

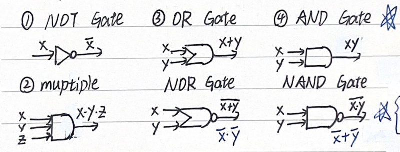
$\overline{x \cdot y} = \bar{x} \cdot \bar{y}$

$\overline{x+y} \ne \bar{x} + \bar{y}$

② muptiple　NOR Gate　NAND Gate

$\overline{x+y} = \bar{x} \cdot \bar{y}$

$\overline{xy} = \bar{x} + \bar{y}$

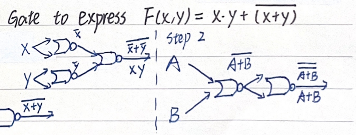


ex: use NOR/~~NAND~~ Gate to express $F(x,y) = x \cdot y + (x+y)$

(1) step1: $x \cdot y = \overline{\bar{x} + \bar{y}}$ ⇒

$\overline{x+y}$ ⇒

step 2



Question 1

The Boolean operator $\oplus$, called the XOR operator, is defined by $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$, and $0 \oplus 0 = 0$. Simplify the below expressions.

a) $x \oplus 0$

b) $x \oplus 1$

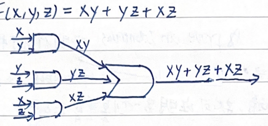
c) $x \oplus x$

d) $x \oplus \bar{x}$

Date.　/　/

ex: design a circuit to test whether at least 2 members vote yes　total 3 members

⇒ $F(x,y,z) = xy + yz + xz$

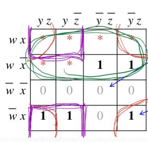


三、Minimization of Circuits

1. K-map P28
2. not care P36

Don’t care conditions

- In some applications, we don’t care certain combinations of input values (indicated as “*” entries in the K-map)
- *Example* 4位二进制编码判断数字≥5
  - We encode a digit (from 0 to 9) by using 4 bits as shown below
  - Given these 4 bits, how to test whether the digit is ≥ 5?
  - First, we draw the K-map as:



digit	w, x, y, z	F
0	0, 0, 0, 0	0
1	0, 0, 0, 1	0
2	0, 0, 1, 0	0
3	0, 0, 1, 1	0
4	0, 1, 0, 0	0
5	0, 1, 0, 1	1
6	0, 1, 1, 0	1
7	0, 1, 1, 1	1
8	1, 0, 0, 0	1
9	1, 0, 0, 1	1

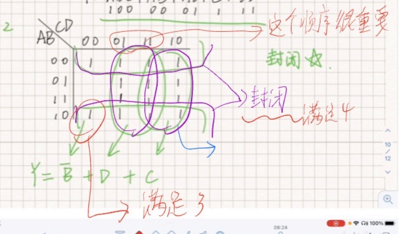
34

试做例题2: $Y = A\bar{B}\bar{C} + \bar{A}B + \bar{A}D + C + BD$

①

$Y = A\bar{B}\bar{C} + \bar{A}B + \bar{A}D + C + BD$

②



这个顺序很重要

封闭

封闭

满足4

满足3

$Y = \bar{B} + D + C$

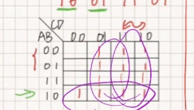
门电路与逻辑代数

卡诺图化简

① 例: $Y = A\bar{B}\bar{C} + \bar{A}C + BC + \bar{C}D$

$A B \bar{C} D$

②



包围圈条件

1. 圈的个数是 2^n　1, 2, 4, 8, 16…
2. 圈尽可能少而大
3. 圈中至少有一个未被圈过的1
4. 卡诺图是封闭的

检验: 化到最简为止

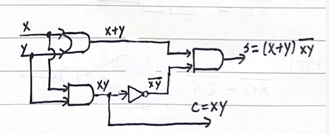
$Y = A\bar{B} + D + C$

* half adder

$$\begin{array}{r} x \\ +\ y \\ \hline c\ s \end{array}$$

$C = x \cdot y$

$S = \bar{x}y + x\bar{y} = (x+y)\overline{xy}$



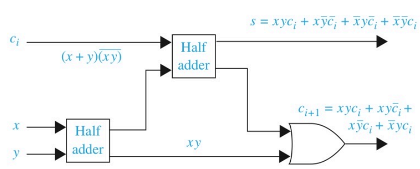


FIGURE 9 A full adder.

(a) 全加器真值表

全加器输入为 A、B (两个1位二进制数) 和 C_{in} (低位进位)，输出为 S (本位和)、C_{out} (高位进位)，真值表如

表格

A	B	C_{in}	S (和)	C_{out} (进位)
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

(b) 本位和 S 的逻辑表达式

S 是 A、B、C_{in} 的三次异或运算，表达式为：

$S = A \oplus B \oplus C_{in}$

也可展开为与或形式：

$S = \overline{A}\overline{B}C_{in} + \overline{A}B\overline{C_{in}} + A\overline{B}\overline{C_{in}} + ABC_{in}$

(c) 进位输出 C_{out} 的逻辑表达式

C_{out} 是“两个输入同时为 1”或“其中一个输入与低位进位同时为 1”的逻辑或，表达式为：

$C_{out} = AB + AC_{in} + BC_{in}$

也可化简为与或非形式：

$C_{out} = (A+B)(A+C_{in})(B+C_{in})$