**1.5：**

**a：1.先用一个乘法箱来计算a\*x**

**2.再使用一个加法箱将ax和b相加，得到ax+b**

**B：1.使用两个加法箱分别将w,x和y,z相加**

**2.再使用一个加法箱将这两个结果相加，得到w+x+y+z**

**3.使用一个乘法箱将总和乘以1/4，得到平均值**

**C：1.这个表达式可以简化为(a+b)^2**

**2.使用一个加法箱来计算a+b**

**3.使用一个乘法箱来计算(a+b)\*(a+b)，即(a+b)^2**

**1.12：**

**A．是算法，是一个明确的步骤，可以按照固定的规律执行**

**B．不是算法，算法是有限的，题目中缺乏明确的步骤和终止条件**

**C．是算法，是一个明确的计算过程**

**D．是算法，有明确的步骤**

**E．是算法，有明确的步骤和终止条件**

**1.13：**

**他们解决问题是相同的，虽然计算机A有减法指令，计算机B没有，但都具有对一个数求负值的指令。B中的减法运算可以转换成加上这个数的相反数，例如:a-b可以转换成a+(-b)。而除了A具有减法指令外，其他指令完全相同。故A和B解决的问题一样多**

**1.22：**

**我认为微结构到电路的转换难度最大：**

**微结构层：设计处理器设计，属于抽象的硬件功能描述 电路层需要将微结构的逻辑功能转化为具体的物理电路，需满足严格的物理约束**

**在这一层需要将抽象的微架构设计转化为具体的电路实现，涉及复杂的电子工程和物理限制，难度大**

**1.23:**

**1.ISA是软件和硬件之间的接口，保持ISA不变可以确保就软件在新硬件上正常运行。Intel公司确保Pentium III的ISA与Pentium II相同，用户升级CPU时无需更换软件**

**2.开发新的ISA需要大量资源和时间，保持ISA不变可以集中资源优化微结构**

**2.11：**

**a. 01100110**

**b.01000000**

**c.00100001**

**d.10000000**

**e.01111111**

**2.17:**

**a. 01+1011=1100 十进制：-4**

**b. 11+01010101=01011000 十进制：88**

**c. 0101+110=1011 十进制：-5**

**d. 01+10=0001+0010=0011 十进制：3**

**2.27：**

**该结果没有问题，LC-3对补码0101010101010101和0011100111001111进行相加得到1000111100100100，说明计算机发生了溢出且是正溢，两个补码均为正数，相加结果超出了2^16-1，导致结果变为负数**

**2.34：**

**a.NOT(1011) OR NOT(1100)=(0100 )OR (0011)=0111**

**b.NOT(1000 AND (1100OR 0101)=NOT(1000 AND 1101)=0111**

**2.37:**

**判断n,m,s的最高位，若n,m的符号位不同，则不溢出，若n,m,s的符号位相同，则不溢出，反之则溢出，程序如下：**

**#include<iostream>  
using namespace std;  
int main(){  
 int n,m,s;  
 cin>>n>>m>>s;  
 int n\_high\_bit=(n&8)>>3;  
 int m\_high\_bit=(m&8)>>3;  
 int s\_high\_bit=(s&8)>>3;  
 int high=n\_high\_bit^m\_high\_bit;  
 if(high||(!high&&n\_high\_bit==s\_high\_bit)){  
 cout<<"1000"<<endl;  
 }  
 else{  
 cout<<"0000"<<endl;  
 }  
}**

**2.39:**

**a.将3.75表示成二进制数11.11**

**即3.75=1\*2^1+1\*2^0+1\*2^-1+1\*2^-2**

**正则化处理后，即为1.111\*2^1**

**符号位为0，表示3.75是一个整数，指数为无符号128，即为10000000，代表实际+1.小数部分省略小数点左边的部分，精度为32.因此，小数部分为11000000000000000000000，由此3.75的IEEE浮点数表示为01000000011000000000000000000000**

**b.将-55(23/64)表示成二进制数为：-110111.010111**

**即-55(23/64)=-(1\*2^5+1\*2^4+0\*2^3+1\*2^2+1\*2^1+1\*2^0+0\*2^-1+1\*2^-2+0\*2^-3+1\*2^-4+1\*2^-5+1\*2^-6)**

**正则化处理后，即为-1.10111010111\*2^5**

**符号位为1，表示一个负数，指数为无符号整数132，即为10000100代表+5，小数部分省略小数点左边的部分为10111010111000000000000，故IEEE浮点数表示为11000010010111010111000000000000**

**2.48：**

**b：111 转换为二进制补码：01101111 十六进制 6F**

**d：-44 转换为二进制补码：11010100 十六进制 D4**

**2.52：**

**十六进制数：x434F4D50**

**无符号二进制数：1000011010011110100110101010000**

**二进制反码形式：10111100101100001010111110101111**

**二进制补码形式：10111100101100001010111110101111**

**IEEE 754浮点数形式：0 10011110 01111001011000010110000**

**ASCII字符串：COMP**

**2.55：**

**a.三位四进制数能表达的最大无符号数值为：63**

**b.n为四进制能表达的最大无符号数值为：4^(n+1)-1 【四的n+1次方-1】**

**c.计算无符号四进制023和221相加的结果：52（十进制）**

**d.十进制数42的四进制表示：222**

**e.四进制数123.3转换为二进制为：011011.11**

**f.四进制数123.3转换为IEEE浮点数表示：0 10000011 11011110000000000000000**

**g.假设一个黑箱函数，该黑箱的输入是一个m为四进制数，输出是一位四进制数字，该黑箱函数的可能的实现数：4^(4^m)**