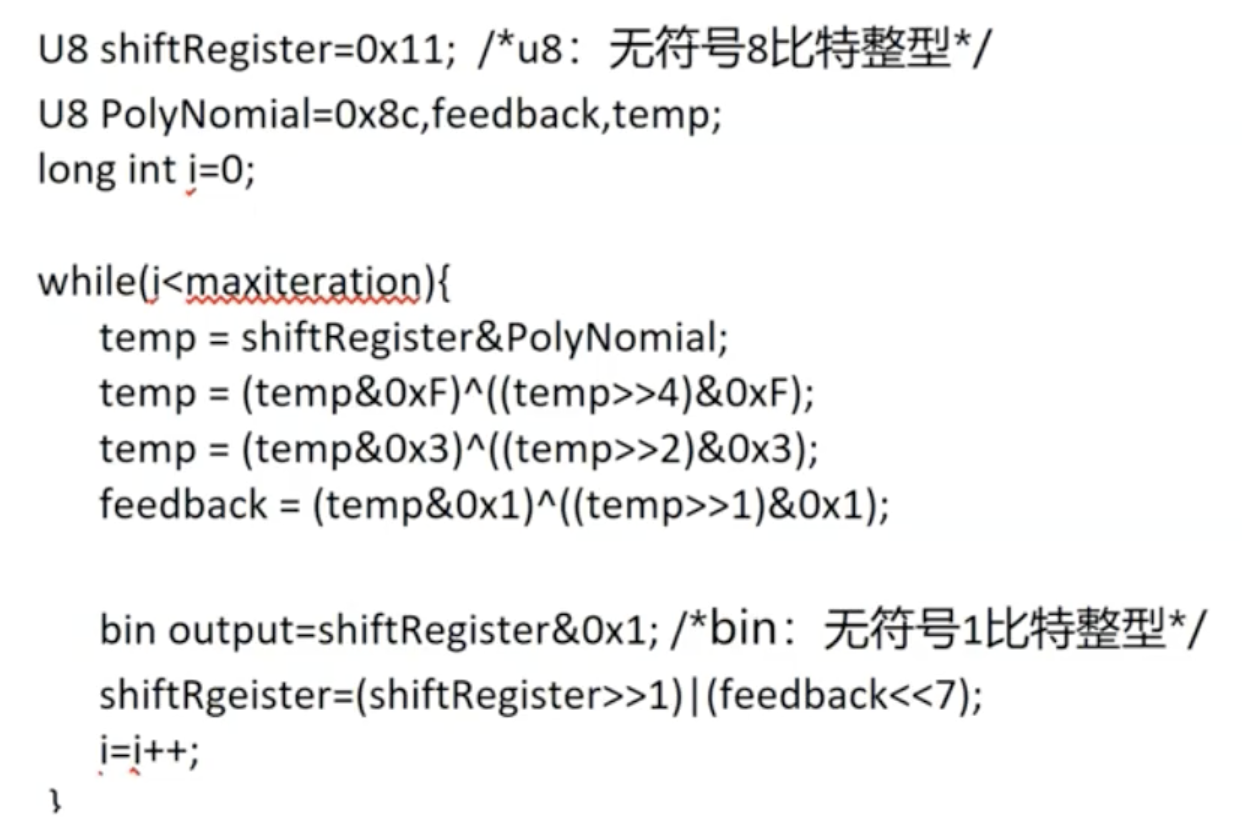
作业一：33级LFSR的实现

学号： 姓名：

要求：

1. 实现以下伪代码
2. 实现33级LSFR
3. 输出从第66个时钟周期后开始
4. 连续输出64比特



大致思路：

1. LSFR为二进制比特操作，因此选择调用中的类，模板类由若干个位（bit）组成，使程序员不必通过位运算就能很方便地访问、修改其中的任意一位。类中有多个比特操作的函数，足够满足本题需求

类大致介绍：

实例化：

右移：

实例中1的个数：

1. 按照伪代码的含义给出代码即可

代码：

#include <iostream>

#include <bitset>

#define N 33

#define maxiteration 130 // 66 + 64

using namespace std;

int main()

{

bitset<N> shiftRegister = 0b000100010001000100010001000100010; // 初始化寄存器的值

bitset<N> PolyNomial = 0b000110011001100110011001100110011; // 初始化多项式的值

long int i = 0;

while(i < maxiteration)

{

if(i >= 66)

cout << shiftRegister[0] << endl; // LFSR运行66个clock后,输出寄存器最右端的值

//cout << shiftRegister.to\_string() << endl;

bitset<N> temp = shiftRegister & PolyNomial; // 多项式与寄存器的值相与

shiftRegister.operator>>= (1); // 寄存器中值右移1位

int feedback = temp.count() % 2; // 计算temp中1的个数,个数取余2，即求出feedback

shiftRegister[N - 1] = feedback; // 更新寄存器最左端的值

i ++ ;

}

cout << "此时寄存器中的值: " << shiftRegister.to\_string() << endl;

return 0;

}

如代码中所示，

将寄存器初值设置为：**000100010001000100010001000100010**

多项式的值设置为：**000110011001100110011001100110011**

对寄存器进行总计130次迭代，从第67次开始输出寄存器右移以前的最低位。

对于feedback的计算，伪代码中的多次相与可以**等价为temp中1的个数对2的余数**，因此只需通过等价运算计算出feedback后填入右移后寄存器的最左端即可。

该程序的输出为：

0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 1 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 0 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 0

最后的寄存器状态：**101111000010001100101011110000100**