题目二

一个程序员，写了如下 C 语言代码。他想在多核处理器上运行该程序，用两个线程，每个线程占一个核。处理器采用顺序一致性协议，变量 X 和 flag 存储在内存中，变量 a 和 b 存储在寄存器中。所有内存初始化为 0。假设每一行 C 代码代表一条指令。

Thread T0	Thread T1
指令 T0.0　X[0] = 1;	指令 T1.0　X[0] = 0;
指令 T0.1　X[0] += 1;	指令 T1.1　flag[0] = 1;
指令 T0.2　while(flag [0]] == 0);	指令 T1.2　b = X[0];
指令 T0.3　a = X[0];	
指令 T0.4　X[0] = a*2;	

1. 变量 a 的最终值可能是多少？ 解释原因

答：
顺序一致性保证每个线程自己的指令顺序执行。跨线程的同步由 flag[0] 来完成。Thread 0 在 T0.2 停留，直到 flag 被 T1.1 置成 1。至少有 3 种不同的顺序一致性顺序：
T1.0->T0.0->T0.1->T0.3　a 的值为 2
T0.0->T1.0->T0.1->T0.3　a 的值为 1
T0.0->T0.1->T1.0->T0.3　a 的值为 0

2. 变量 X[0]的最终值可能是多少？解释原因。

答：
X[0]的最终值是 a 值的 2 倍，分别为 4, 2, 0

3. 变量 b 的最终值可能是多少？解释原因。

答：
变量 b 的最终值可能是 0, 1, 2, 4。
T1.2 的顺序并没有约束，因此变量 b 可能在任意指令位置执行，即可能等于 X[0]的任何可能值。

4. 假设该程序员想让变量 a 和 b 的取值在程序执行完毕时是相同的，那么在保留 T1.1 和 T0.2 两条指令的前提下，需要对原始程序如何做最小的改动（填写下表）？（提示：可以利用更多的 flags）

答：

需要保证指令 a=X[0]和指令 b=X[0]之间不能有其它修改变量 X[0]的指令。

Thread T0	**Thread T1**
T0.0 X[0] = 1;	T1.0 X[0] = 0;
T0.1 X[0] += 1;	T1.1 flag[0] = 1;
T0.2 while(flag[0] == 0);	T1.2 while(flag[1] == 0);
T0.3 a = X[0];	T1.3 b = X[0];
T0.4 flag[1] = 1;//可以放在 T0.1 和 T0.3 间	T1.4 flag[2] = 1;
T0.5 while(flag[2] == 0);	
T0.6 X[0] = a*2;	