(a) The backward slice for load L2 is as follows:
R2=R7*R4
R4=R3*R4
R5=R4*2
Load R3,0(R5) ==> load L1 (always a hit)
R2=R3+R5
Load R1,0(R2) ==> load L2

The backward slice contains 5 instructions (not counting load L2) and hence the backward slice takes 5 cycles to execute and issue a prefetch. Then 10 cycles later data is guaranteed to be available for load L2. However there are only 9 instructions ahead of load L2 in the code segment. Hence the best one can do is to initiate the slice computation at the start of the code segment. The slice takes 5 cycles to initiate the prefetch. Four cycles later load L2 is executed but the prefetch is already issued and pending and hence four cycles of the miss penalty have been saved.

(b) If load L1 has a 20% miss probability it takes 10 additional cycles to satisfy that miss. Hence the average latency of load L1 is 1*0.8+10*0.2=2.8 cycles. Hence the execution of the backward slice must be initiated at least 16.8 cycles ahead of load L2 in order to completely hide its latency. However, a slice can only be initiated 9 cycles ahead at best, of which 6.8 cycles are needed for slice computation, leaving just 2.2 cycles to cover the latency of load L2. Therefore it is still beneficial to fire the backward slice for load L2 at the beginning of the code segment even if the miss rate of load L1 is 20%.

Problem 8.8

For simplicity assume that each thread executes 200 instructions.
(a) The CPI is (200+10+50)/200 = 1.3

(b) Cache access latencies are hidden because of the total overlap of thread executions with cache misses. The CPI improves to 1.

(c) With a switching overhead of 5 cycles the CPI is now:
(100+5+100+5)/200 = 1.05.
This increase reflects the overhead of switching threads.

(d) CPI(T1) = (100+50+100+500)/200 = 3.75
CPI(T2) = (100+50+100+500)/200 = 3.75
The CPI of the overall machine is ~~3.75~~ as well. Check the solution slides

Problem 8.9

In this problem we have two variables: the number of cores (P) and the number of L2 cache banks (N). We need to find a configuration that achieves the maximum speedup with respect to the base machine with three cores and nine L2 cache banks, given that area is expanded by 4x. To obtain this optimum, we derive the following equations.

a. Area equation:
Since each core occupies one unit of area and each L2 cache bank occupies three units of area, the total area for the base machine is 30 (i.e., 3 + 27) units of area. With the extra real estate provided for the new machine, we have four times the area of the base machine, which is 120 (i.e., 30 * 4) units of area. Thus the number of cores (P) and the number of L2 cache banks (N) in the new