Multi Processor: threads multiprocessing

1. Only one thread can be active at a time on each core we have. So, putting thread X to core 1 and Thread Y to core 2 we can separate each core from each other. On core 1, we cannot run A4 before A3 finishes and A1 and A3 can't run simultaneously. A good execution would be A3 and A2, A1 and A4, A1 then A1 because we wanted to run A3 as soon as possible. A1 takes 3 cycles and A3 takes 1 cycle and they can't run together so this seems like the more progressive way to run these executions. On Core 2, we run similarly: B2 and B4, B1 and B4, B1 and B3 which uses 2 FUs every cycle. We need 4 cycles in total and we have 16 issues slots in total. 12 of them are used while the others remain wasted.
2. There is no better execution cycle, so we need 4 cycles. But the number of issues slots doubles and we have 32 now. 12 are still being used while the other 20 are wasting away.
3. The CPU must change the active thread when an active operation executes. A good execution cycle would be A2 and A3, B2 and B4, B4, A1 and A4, A1, A1, B1 and B3. With this we need 8 cycles with 16 issue slots, and we are using 12 while the other 4 waste.