Lecture 1

Exercise 1: Ok!

Exercise 2: SIMD - Single Instruction, Multiple Data: The instruction is executed at each clock cycle, but on different inputs. Synchronous and deterministic execution. Think GPU. MIMD - Multiple Instruction, Multiple Data: Every processor performs any arbitrary instruction, on any input. Does not need to be synchronous or deterministic. Think multi core CPU. SPMD - Single Program, Multiple Data: A programming pattern for MIMDs where the logic for which of the different instructions to be executed is contained, and triggered by the data.

Exercise 3: The largest relative change in clock rate between generation is the jump from 200 MHz to 2000 MHz (2 GHz) between 1997 and 2001: A difference of 1 800 MHz. The largest jump in power consumption is between the same generations: Going from 29.1 W to 75.3 W, a difference of 46.2 W. The latest version presented in the table has 213.4 times as much clock rate as the first generation, while only needing 28.8 times the power.

Lecture 2

Exercise 4: If all philosophers start at the *same time*, and think for same amount of time, they will all pick up the fork to their left at the same moment. This means the fork to their right will be busy, and none of them can move on to the third step.

Exercise 5: One solution would be to add a condition to the third step: If the right fork is busy, release the left fork, wait for a moment, and try from the second step (picking up the left fork) again. And as the question indicates, in some cases this might mean that one of the philisophers is unlucky and will never manage to get both forks because the neighbouring philosophers keep taking them.

Exercise 6: The single shared memory processor system will have 20 - 1 = 19 GB available. The cluster has 54 - 51 = 15 GB available. So the shared memory system has 4 GB more available memeory.

Exercise 7:

Exercise 8: With hybrid systems one could more easily scale up the performance of the system by adding 'components'. On for example a pure shared memory system the bandwidth to the memory eventually becomes a bottleneck as you add processors, and would have to be imcreased, which might be complicated/expensive. On the other hand, just as with pure DMMPs, making programs run efficiently with the different types of components might be a hassle, and one must ask oneself whether it is worth the additional maintenance of software running on the system, or even the system itself when it becomes complex.