CPSC 313

06W Term 2

Problem Set #8 - Solution

1. SRAM stands for Static Random Access Memory. It can be typically found in caches, on or off the core. It is fast and persistent, but it requires more transistors so is more expensive per unit of memory that Dynamic RAM.

2. DRAM stands for Dynamic Random Access Memory. It can be typically found in the Main Memory that is attached to the motherboard. It consists of only one transistor and one capacitor per bit, so it is cheaper and can be made denser (less surface area per bit) than SRAM. However, DRAM is slower compared to SRAM, and requires periodical recharge.

3. One reason that is indicated by question 1 and 2 is that cache and memory have different prices and speeds. Therefore, using a cache/memory hierarchy is able to strike the best balance between them by keeping the small but frequently-accessed data/code in the cache, and putting the other part in the low-speed but cheap, enormous memory.

   The other reason that enables the cache-memory hierarchy is the existence of spatial and temporal locality. If there was no locality, then caching would be ineffective.

4. In a program with good spatial locality, if a memory location is referenced, then the program is likely to reference a nearby memory location in the near future. Example:

```
for (i=0; i<n; i++)
  C[i] = 1;
```

   Spatial locality enables cache prefetching, i.e., loading nearby data (a whole block at a time) from memory into cache every time the cpu issues a memory read/write.

5. In a program with good temporal locality, a memory location that is referenced once is likely to be referenced again multiple times in the near future. Example:

```
for (i = 0; i<n; i++)
  j = (j+1)*2^i;
```

   Temporal locality justifies the fact that a cache keeps recently-access data for a while so that the later accesses will hit in the cache instead of fetching the data/code from memory again.

6.

Cache	m	C	B	E	S	t	s	b
1.	32	1024	4	4	64	24	6	2
2.	32	1024	4	256	1	30	0	2
3.	32	1024	8	1	128	22	7	3
4.	32	1024	8	128	1	29	0	3
5.	32	1024	32	1	32	22	5	5
5.	32	1024	32	4	8	24	3	5

7. Set 1 contains the blocks with tag 0x45 and 0x38. Shifting these tags left 5 bit positions and or-ing in 0x4 (the index 1 left shifted 2 bit positions) in gives the address of the first byte of the data in these blocks, or 0x8a4 and 0x704, so the 8 addresses that will hit are 0x8a4, 0x8a5, 0x8a6, 0x8a7, 0x704, 0x705, 0x706, 0x707.

   Set 6 contains the block with tag 0x91, similar processing yields 0x1238, 0x1239, 0x123a, 0x123b.

8. A. Total writes: 1024
   B. Cache misses: 128
   C. Miss rage: 12.5%

9. A. Total writes: 1024
   B. Cache misses: 256
   C. Miss rage: 25%

10. A. Total writes: 1024
    B. Cache misses: 256
    C. Miss rage: 25%