1. (10 point: 1 point each) The following are true/false questions. Please write either T or F on the line.

1. \_\_\_\_\_\_ Capacity misses can be eliminated by increasing cache associativity.
2. \_\_\_\_\_\_ Write-through caches are typically no-write-allocate, whereas write-back caches are typically write-allocate.
3. \_\_\_\_\_\_ Potential memory aliasing may be an optimization blocker.
4. \_\_\_\_\_\_ When a data item is written repeatedly, a write-through cache probably will have better performance than a write-back cache.
5. \_\_\_\_\_\_ The clock speed for a pipelined architecture is determined by the speed of the fastest stage of the pipeline.
6. \_\_\_\_\_\_ A pipelined machine’s ISA should be logically identical to the non-pipelined version.
7. \_\_\_\_\_\_ Load/use is an example of a control hazard in a pipeline
8. \_\_\_\_\_\_ Forwarding eliminates the needs for bubbles in the Y86 pipeline.
9. \_\_\_\_\_\_ Locality means that programs tend to reference data instructions near those recently referenced.
10. \_\_\_\_\_ Flash Memory is volatile, but DRAM is not.

2. (10 points) The stages of our Y86 pipeline are: **Fetch, Decode, Execute, Memory,** and **Write Back.** For each of the following, state at which stage is most likely to occur. (If there are multiple answers, state the best answer)

1. \_\_\_\_\_\_ Value read in from data memory location
2. \_\_\_\_\_\_ Value written to data memory location
3. \_\_\_\_\_\_ Effective address computed including displacement
4. \_\_\_\_\_\_ Instruction split into fields
5. \_\_\_\_\_\_ Value read from register
6. \_\_\_\_\_\_ Value written to register
7. \_\_\_\_\_\_ Instruction read from instruction memory
8. \_\_\_\_\_\_ Values added in ALU
9. \_\_\_\_\_\_ Branch condition is computed
10. \_\_\_\_\_\_ Condition codes are computed

3. (6 points) Given the following C structures, indentify the offset (not already supplied) for each listed object within the structure. Assume the structure starts at address zero. Use our standard gcc compiler alignment rules.

struct {

short unsigned int w;

char q[3];

double f;

unsigned int l;

union {

int a[2];

int \*p;

char r[9];

} our\_union;

int b;

} our\_structure;

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Offset | Variable | Offset |
| w | 0 | r |  |
| q | 2 | p |  |
| q[1] | 3 | a[1] |  |
| f | 8 | r[7] |  |
| l |  | b |  |

4. (5 Points) Assume we have a 2-way set associative 1024-byte data cache with 16-byte blocks. Assume the machine used 32-bit addresses and the memory is byte addressable. Determine the following cache parameters.

* The block offset takes \_\_\_\_\_\_ bits
* The set index takes \_\_\_\_\_\_ bits
* The tag takes \_\_\_\_\_\_ bits
* There are \_\_\_\_\_\_ total sets
* There are \_\_\_\_\_\_ total cache lines

5. (3 points) A hard drive for a laptop computer has the parameters given in the table below:

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Rotational speed | 6000rpm |
| Average seek time | 10ms |
| Transfer rate | 10M bytes/sec |

Compute the average amount of time required to read or write 100k bytes. Assume that the entire transfer requires only one seek and latency penalty. Show your work, but write your answer on the line.

Answer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. (4 points) The contents of an associative cache is shown below. This cache has four-byte blocks and 32-bit addresses. Tags and sets are shown in binary, and data values in hex. The “...” should be replaced by an appropriate number of 0’s.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Set | V | Tag | 0 | 1 | 2 | 3 |
| 0 | 1 | 11110...011110000 | 0x41 | 0x33 | 0x22 | 0x11 |
|  | 1 | 10000...010110010 | 0xAA | 0xBB | 0xCC | 0xDD |
| 1 | 1 | 10000...011110000 | 0x01 | 0x02 | 0x03 | 0x04 |
|  | 1 | 10000...010000000 | 0x12 | 0xAB | 0x34 | 0xEF |

What are the addresses that are cached in each block? Write the range in hexadecimal using all eight digits, e.g. 0x00000000-0x000000AC).

Line A: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Line B: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Line C: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Line D: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Questions 7-22 are multiple choice. Write the letter of the BEST answer in the space. Please write your answer in UPPERCASE.**

7. \_\_\_\_\_\_ (2 points) Pipelining a processor implementation probably *won’t* do which of the following:

1. Increase throughput;
2. Decrease latency;
3. Allow a faster clock rate;
4. All of the above probably will happen

8. \_\_\_\_\_\_ (2 points) The pipelined Y86 tries to predict the PC, except for:

1. Calls
2. Returns
3. Unconditional jumps
4. Conditional jumps
5. Moves

9. \_\_\_\_\_\_ (2 points) The L1 cache on a high-end processor is most likely to use which technology?

1. SRAM
2. DRAM
3. Flash
4. Magnetic disk

10. \_\_\_\_\_\_ (2 points) Which of the following is not true of a solid state disk (SSD)?

1. Random writes are much slower than sequential writes
2. Writing a page requires erasing a block
3. There are no moving parts
4. Blocks wear out after many reads / writes
5. All of the above are true

11. \_\_\_\_\_\_ (2 points) The memory hierarchy has each of the following characteristics except:

1. Smaller, faster memories are “closer” to the processor;
2. Faster caches use more expensive technology
3. Caching tends to work because of locality
4. A read miss at one cache level *k* requires reading from level k + 1
5. All of the above are true

12. \_\_\_\_\_\_ (2 points) In a system using caching, which of these is probably not a good strategy to improve average memory access time?

1. Reducing the miss rate;
2. Reducing the hit rate;
3. Reducing the miss penalty;
4. Reducing the hit time;

13. \_\_\_\_\_\_ (2 points) *Reduction in strength* refers to:

1. Replacing costly operations with cheaper ones;
2. Letting yourself go;
3. Moving code of out of loops;
4. Handling multiple elements within one loop iteration.

14. \_\_\_\_\_\_ (2 points) A compiler typically applies weak optimization around procedure calls because:

1. Procedures may have side-effects
2. Call results may depend on global data
3. Inter-procedure analysis is expensive
4. All of the above

15. \_\_\_\_\_\_ (2 points) Suppose you have a cache with capacity of 32768 (= 2^15) bytes with 32-byte blocks. Assume 8 bits are used to select the set. What is the associativity of the cache?

1. The cache is direct-mapped.
2. The cache is two-way set associative.
3. The cache is four-way set associative.
4. The cache is eight-way set associative.
5. None of the above.

16. \_\_\_\_\_\_ (2 points) which of the following is not always true of a write in a cache:

1. In a write-back cache, the dirty bit is set;
2. In a write-through cache, bus traffic occurs;
3. A write miss requires evicting a line from the cache.

17. \_\_\_\_\_\_ (2 points) Deciding which line to evict on a cache miss requires consulting:

1. The eviction policy
2. The placement policy
3. The replacement policy
4. The foreign policy

18. \_\_\_\_\_\_ (2 points) All of the following are involved in the computation of CPI (cycles-per-instruction) except:

1. Load penalties;
2. Branch misprediction penalties;
3. Return penalties;
4. All of the above are relevant.

19. \_\_\_\_\_\_ (2 points) Consider the following C code

X = 1000; y = 3000;

\*q = y;

\*p = x;

1 = \*q

After execution of this code, what is the value of t1?

1. 1000
2. 2000
3. 3000
4. We can’t be sure

20. \_\_\_\_\_\_ (2 Points) A hard disk for a laptop computer has following characteristics:

* 128 bytes/sector
* 200 sectors/track (on average)
* 1,000 tracks/surface
* 2 surfaces/platter
* 4 platters

The capacity for this disk is closest to:

1. 2GB B. 1GB C. 200MB D. 20GB

21.\_\_\_\_\_\_ (2 points) Which of the following (if any) is probably not useful when attempting to improve program performance?

1. Profiling
2. Code motion
3. Loop unrolling
4. Eliminating optimization blockers
5. All of these can be useful

22.\_\_\_\_\_\_(2 points) All of the following are differences between the sequential and pipelined Y86 implementations except.

1. The pipelined version has more sequential logic;
2. The clock speed of the sequential version is probably slower;
3. The final PC after and instruction may differ;
4. There are no control hazards in the sequential versions;
5. All if these are differences

1.

1. F
2. T
3. T
4. F
5. F
6. T
7. F
8. F
9. T
10. F

2.

1. M
2. M
3. E
4. F
5. D
6. W
7. F
8. E
9. E
10. E

3.

4. 4

5

23

32

64

5. 25ms

6. 0xF0000780 - 0xF0000783

0x80000590 - 0x80000593

0x80000784 - 0x80000787

0x80000400 - 0x80000407

7. B

8. B

9.

10.

11. E

12.

13.

14.

15.

16.

17.

18.

19.

20.

21.

22.