Comp 2825 - Computer Architecture

Exercise Solutions – Week 2

Exercise 1 – Memory Organization

1. **B – 2048 bytes**

   Each cell holds 16 **bits**, and since there are 8 bits in a byte this means the cell holds 2 **bytes**. Since there are 1024 cells, the total capacity of the memory is 2 bytes x 1024 cells = 2048 bytes.

   Another way to arrive at the answer: The total capacity of the memory is 16 bits/cell x 1024 cells = 16384 bits. Since there are 8 bits in a byte, there are 16384 / 8 = 2048 bytes.

2. **B – 10 bits**

   There are 1024 cells in the memory, so the address needs to be large enough to be able to represent 1024 unique combinations of 1 and 0 bits. 2^{10} = 1024, so we know that a number with 10 bits can have 1024 combinations of 1's and 0's.

   Remember the handy powers of 2: 2^{10} = 1K, 2^{20} = 1M, 2^{30} = 1G

Exercise 2 – Byte Ordering

1. **A – Little-Endian**

   The low-order byte (containing hex 64) is stored in the memory cell with the lowest address (106).

2. **B – Big-Endian**

   The high-order byte (containing 0) is stored in the memory cell with the lowest address (0).

3. **A – Little-Endian**

   It's important to pay attention to the address number. This memory diagram has the memory cells reversed so that the 2-byte number appears high-order byte first even though it's in Little Endian format. This is a common way to show memory on Little-Endian systems. What's important is that the low-order byte (containing hex 64) is in the memory cell with the lowest address (416).

Exercise 3 – Row / Column Parity

1. **C – The Column 1 Parity Bit**

   There is an error in Column 1 because there is only one 1-bit in that column, which is not an even number. But there are no errors in the rows (both rows have an even number of 1-bits). If the error in column 1 was in one of the data bits then one of the rows would have bad parity. Since it doesn't the problem must be with the Column 1 parity bit itself.

2. **B – The Column 1 / Row 2 Data Bit**

   There is an error in Column 1 because the number of 1-bits is odd, and there's also an error in Row 2. So the bad bit must the one that's in both the bad row and the bad column.

Exercise 4 – Correcting Errors in a 4-bit Hamming Code

1. **D – The lower-left Data Bit**

   The upper-left and lower circles have an odd number of "1" bits – so the problem must lie with a bit that is common to both those circles. That narrows it down to the "D" and "E" bits – but the "E" bit is also part of the upper-right circle which has correct parity. So it the problem must lie with the "D" bit.

2. **E – The centre Data Bit**

   All three circles have odd (bad) parity. The only bit which could go bad and affect all three parity circles is the centre "E" bit.

3. **C – The upper-right Parity Bit**

   The only circle that has an odd number of bits (bad parity) is the upper-right circle. If any of the data bits in this circle were bad then it would cause errors in the other circles as well, but they are OK. The only bit that could go bad and not affect the other parity circles is the upper-right parity bit.

Exercise 5 – Layout of a Hamming Codeword for 16 data bits

D – To hold 16 data bit we need 16 of the "blue" (non-parity) bit positions, and the bit numbers for computing Hamming parity sums must start with "1", not "0".

Exercise 6 – Hamming Codeword Validation

1. **B – Incorrect**

   There are seven "1" bits in the P1 row. Since there are supposed to be an even number of "1" bits, this means that one of the bits must be incorrect.

2. **A – Correct**

   There are four "1" bits in the P2 row. Since that's an even number, it means all the bits in this row are correct.

3. **B – Bit 5**

   The P4 row is incorrect because it has five "1" bits (an odd number).
   The P8 row is correct because it has two "1" bits (an even number).
   The P16 row is correct because it has four "1" bits (an even number).
   So the incorrect rows are P1 and P4.
   Therefore the bad bit must be 1+4 = 5. This is the only bit which is common to the P1 and P4 rows and none of the other rows. So if we switch this bit from a "0" to a "1" it will correct the P1 and P4 parity sums and leave the other parity sums correct as well.

Exercise 7 – Hamming Distance

C – The minimum number of bits that are different between valid codewords

Remember that the "codeword" includes both the data bits and the parity bits.

Exercise 8 – Cache Access Times

1. **A – 20%**

   80% of reads are found in the cache and returned to the CPU immediately – that leaves 20% which go on to main memory (of which 100% are found there).

2. **B – 8 ns**

   Every read takes 2ns to search the cache.
   20% of the reads take an additional 30ns to access from main memory

```
  100% x 2 ns  = 2 ns
+ 20% x 30 ns = 6 ns
                8 ns
```

Exercise 9 – Cache Access Times

1. **C – 100%**

   Every read goes to the L1 cache first.

2. **A – 15%**

   85% of all reads are found in the L1 cache, so only the leftovers (100% -85% = 15%) go on to the L2 cache. 15% is the "miss rate" of the L1 cache.

3. **B – 0.75%**

   Of the 15% of reads which reach the L2 cache, only the ones that aren't found go on to main memory. The ones that aren't found are 100% - 95% = 5% (the miss rate of the L2 cache). The 5% miss rate x the 15% that reach L2 gives us (0.05 x 0.15 = 0.0075, or 0.75%) that reach main memory. (i.e., L2's miss rate times L2's % of reads that reach that level in the table below).

4. **C – 3.2 ns**

   Adding "Miss rate" and "% of reads that reach this cache level" can help to solve the problem:

	Hit Rate	Access Time	Miss Rate	% of reads @ this level	% of reads X access time:
Level 1 Cache	85%	2ns	15%	100%	100% x 2ns = 2.0 ns
Level 2 Cache	95%	6ns	5%	15%	+ 15% x 6ns = 0.9ns
Main Memory	100%	40ns	0%	0.75%	+ 0.75% x 40ns = 0.3ns
					Total: 3.2ns

Note that 0.75% x 40ns is calculated as 0.0075 x 40ns.

Exercise 10 – Circuit vs. Algebraic Expressions

H – NOT((A OR B) AND C)

A and B go into an OR gate, which gives (A+B) as a result.
The NAND gate has (A+B) as one input, and C as the other.

Exercise 11 – Boolean Logic Circuit

A – Note that all of the other truth tables show a TRUE output in one or more of the first four cases where the door is closed (D=0)

Exercise 12 – Boolean Logic Circuit

B – According to the truth table, none of the cases has NOT-D, so that rules out choices C and D
The correct Sum-of-Products form has the three OR clauses (sums) joined by ANDs (products)

Exercise 13 – Boolean Logic Circuit

C – According to the Boolean expression, none of the cases has NOT-D. The AND gate input identified by "C" should be connected to D, not to NOT-D.

Exercise 14 – Circuit Equivalence

C – In the first pair of tables, the result expressions on both sides are incorrect.
In the second pair of tables, the A+B / A•B results are incorrect.