**Task 1. A) Convert student number to binary.**

|  |  |  |  |
| --- | --- | --- | --- |
| LSB 1695074  2 3390149  2  ---------  13  12  ----------  19  18  ----------  10  10  ----------  01  02  -----------  14  14  ------------  9  2  -------------  R 1 | 847537  2 1695074  16  ---------  9  8  ----------  15  14  ----------  10  10  ----------  7  6  ----------  14  14  ----------  0 | 42376 8  2 847537 8  ---------  4  4  ----------  7  6  ----------  15  14  ----------  13  12  ----------  17  16  ----------  R 1 | 211884  2 423768 4  ---------  2  2  ----------  3  2  ----------  17  16  ----------  16  16  ----------  8  8  ----------  0 |
| 105942  2 211884 2  ---------  1  2  ----------  11  10  ----------  18  9  ----------  8  8  ----------  4  4  ----------  0 | 52971    2 105942 10  ---------  5  4  ----------  19  18  ----------  14  14  ----------  2  2  ----------  0 | 26485    2 52971 4  ---------  12  12  ----------  9  8  ----------  17  16  ----------  11  10  ----------  R 1 | 13242    2 26485  2  ---------  6  6  ----------  4  4  ----------  8  8  ----------  5  4  ----------  R 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6621    2 13242  12  ---------  12  12  ----------  4  4  ----------  2  2  ----------  0 | 3310    2 6621  6  ---------  6  6  ----------  2  2  ----------  1  2  ----------  R 1 | 1655    2 3310  2  ---------  13  12  ----------  11  10  ----------  10  10  ----------  0 | 827    2 1655  16  ---------  5  4  ----------  15  14  ---------  R 1 | 413    2 827  8  ---------  2  2  ----------  7  6  ---------  R 1 |
| 206    2 413  4  ---------  1  2  ----------  13  12  ---------  R 1 | 103    2 206  2  ---------  0  2  ----------  6  6  ---------  R 0 | 51    2 103  10  ---------  3  2  ---------  R 1 | 25    2 51  4  ---------  11  10  ---------  R1 | 12    2 25  2  ---------  5  4  ---------  R1 |
| MSB  6    2 12  12  ---------    R 0 | 6/2 = 3 R 0  3/2 = 1 R 1  1/2 = 0 R 1 | **0011 0011 1011 1010 1100 0101**  **This is the equivalent in binary for the student number 3390149.** | | |

**Task 1. B) Convert binary string to octal.**

**Octal split result into groups of 3, each set goes from 0 to 7.**

**001 100 111 011 101 011 000 101**

Set ↓ Position 🡪 Result ↓

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **2^2** | **1^1** | **0^0** |  |
|  |  |  |  |  |
| **1** | 0 | 0 | 1 | 1 |
| **2** | 1 | 0 | 0 | 4 |
| **3** | 1 | 1 | 1 | 7 |
| **4** | 0 | 1 | 1 | 3 |
| **5** | 1 | 0 | 1 | 5 |
| **6** | 0 | 1 | 1 | 3 |
| **7** | 0 | 0 | 0 | 0 |
| **8** | 1 | 0 | 1 | 5 |

**Task 1. B) Convert binary string to Hexadecimal.**

**Hexadecimal result is split into groups of 4 with possible values of 0-15 = 0 – F.**

**0011 0011 1011 1010 1100 0101**

Set ↓ Position 🡪 Result ↓

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **2^3** | **2^2** | **2^1** | **2^0** |  |
|  |  |  |  |  |  |
| **1** | 0 | 0 | 1 | 1 | 3 |
| **2** | 0 | 0 | 1 | 1 | 3 |
| **3** | 1 | 0 | 1 | 1 | B |
| **4** | 1 | 0 | 1 | 0 | A |
| **5** | 1 | 1 | 0 | 0 | C |
| **6** | 0 | 1 | 0 | 1 | 5 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Task 1. C + D) Convert student number to base 12, Add 24 to the result**

**Base 12 conversion**

3390149/12 = Subtracting original number by the position values and how many times it will fit into the initial or remaining value.

1 1 7 5 A 8 5

12^6 12^5 12^4 12^3 12^12 12^1 12^0

2985984 248832 20736 1728 144 12 1

This method calculates 12 powers and for this method the greatest value that can go into the number to be divided is less than the value to be divided. For each of the values is subtracted and the amount of times the value can go into the original value is recorded.

**Is the same as….**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Multiply rounded value** |  |  |
|  |  | **of second column by 12 to get** |  |  |
|  |  | **remainder by subtracting with** |  |  |
| 1st column divisible by 12 | | **column 1** | Remainder | |
| 3390149 | 282512 | 3390144 | 5 |  |
| 282512 | 23542 | 282504 | 8 |  |
| 23542 | 1961 | 23532 | 10 | A |
| 1961 | 163 | 1956 | 5 |  |
| 163 | 13 | 156 | 7 |  |
| 13 | 1 | 12 | 1 |  |
| 1 | 0 | 0 | 1 |  |
|  |  |  |  |  |
| **Adding 24** |  |  | Remainder | |
| 3390173 | 282514 | 3390168 | 5 |  |
| 282514 | 23542 | 282504 | 10 | A |
| 23542 | 1961 | 23532 | 10 | A |
| 1961 | 163 | 1956 | 5 |  |
| 163 | 13 | 156 | 7 |  |
| 13 | 1 | 12 | 1 |  |
| 1 | 0 | 0 | 1 |  |

Adding 24 can be divided by twelve 2 times because being the Base 12 number system and will therefore add 2 onto the 1 position to effectively make the new value 1175AA5. The reason this is easy to calculate is the number is divisible by 12 and therefore the result can be placed into its position based on its value without doing any in-depth calculations.

**Task 1. E) Convert student number to base 26 letters instead of numbers,**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name in Base 26 format** | | |  | |  |  |  |
|  | | |  | |  |  |  |
| First Name Ben first letter less than 2 (2 letters) | | | EE | |  |  |  |
| Last Name first 3 letters | | | TOM | |  |  |  |
|  | | | E = 4 | |  |  |  |
|  | | | T = 19 | |  |  |  |
|  | | | O = 14 | |  |  |  |
|  | | | M = 12 | |  |  |  |
| Convert the two sets to Base 10 | | |  | |  |  |  |
| **Position for Base 26** | | | **2** | | **1** | **0** |  |
|  | | | 26 \* 26 | | 26 \* 1 | 1 |  |
| Value | | | 676 | | 26 | 1 |  |
|  | | |  | |  |  |  |
| **Position for set 1** | | |  | | **4** | **4** |  |
|  | | |  | | 26 \* 4 | 4 \* 1 |  |
| Value | | |  | | 104 | 4 | 108 |
|  | | |  | |  |  |  |
| **Position for set 2** | | | **19** | | **14** | **12** |  |
|  | | | 19 \* 676 | | 14 \* 26 | 12 \* 1 |  |
| Value | | | 12844 | | 364 | 12 | 13220 |
| **Total in Base 10** | | |  | |  |  | **13328** |
| **Base 26** | | | **T** | | **S** | **Q** |  |
| Take remainders list from right to left  Until number is an integer or find values of positions and subtract from each until 0 | | | 19 | | 18 | 16 |  |
|  | | | 19 \* 676 | | 18 \* 26 | 16 \* 1 |  |
|  | | | 12844 | | 468 | 16 | 13328 |
|  | | |  | |  |  |  |
|  |  | Remainder | |
| 13328 / 26 | 512 | **16** | |
| 512 / 26 | 494 | **18** | |
| 494 / 26 | **19** | 0 | |

Finding value by dividing by 26 and obtaining remainder until value of the remainder reaches 0.

**Task 2 - Binary Addition**

**A + B). Last 3 numbers for student number: 149 converting each digit to binary and adding result.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Base 10 to Base 2 | |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  | Number 149 | |  |  |  |  |
|  |  | **1** | **0001** |  |  |  |  |
|  |  | **4** | **0100** |  |  |  |  |
|  |  | **9** | **1001** |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | **Overflow** | **Summing the values in binary for validation to 4-bit arithmetic** | | | | | |
| **Position** |  | 3 | 2 | 1 | 0 |  |  |
|  |  |  |  |  |  |  |  |
| **Carry** |  |  |  | 1 |  |  |  |
|  |  | **0** | **0** | **0** | **1** |  |  |
|  |  | **0** | **1** | **0** | **0** |  |  |
|  |  | **1** | **0** | **0** | **1** |  |  |
| **Binary** |  | 1 | 1 | 1 | 0 |  |  |
| **Value** |  | 8 | 4 | 2 | 0 |  |  |
| **Total** |  | **14** |  |  |  |  |  |

**C).** **Binary result to Base 4**

|  |  |  |
| --- | --- | --- |
| **Number 14** | Binary to Base 4 |  |
| **Position** | 1 | 0 |
| **Value** | 4 | 1 |
|  | 3 \* 4 = 14 / 4 = 3 R 2 | 2 \* 1 = 2/1 = 2 |
| **Result** | 3 | 2 |
|  | Position 1 and 0 | Position 1 only |
| **Base 2** | 11 | 10 |
| **Base 4** | 3 | 2 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| A | B | C |  | /C | (B + / C) | /A | /A.(B+/C) |
|  |  |  |  |  |  |  |  |
| 0 | 0 | 0 |  | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 |  | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 |  | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 |  | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 |  | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 |  | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 |  | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 |  | 0 | 1 | 0 | 0 |

**Task 3. A) Truth tables**

**The following identities /A .(B +/C) = (/A+B).(/A+/C) are FALSE.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  | /A | (/A+B) | /C | (/A + /C) | (/A+B) . (/A + /C) |
|  |  |  |  |  |  |  |  |  |
| 0 | 0 | 0 |  | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 |  | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 |  | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 |  | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 |  | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 |  | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 |  | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 |  | 0 | 1 | 0 | 0 | 0 |

**B) The following /(A.B.C) = /A . /B . /C identities are also FALSE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C |  | (A.B.C) | /(A.B.C) |
|  |  |  |  |  |  |
| 0 | 0 | 0 |  | 0 | 1 |
| 0 | 0 | 1 |  | 0 | 1 |
| 0 | 1 | 0 |  | 0 | 1 |
| 0 | 1 | 1 |  | 0 | 1 |
| 1 | 0 | 0 |  | 0 | 1 |
| 1 | 0 | 1 |  | 0 | 1 |
| 1 | 1 | 0 |  | 0 | 1 |
| 1 | 1 | 1 |  | 1 | 0 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  | /A | /B | /C | /A./B./C |
|  |  |  |  |  |  |  |  |
| 0 | 0 | 0 |  | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 |  | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 |  | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 |  | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 |  | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 |  | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 |  | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 |  | 0 | 0 | 0 | 0 |

**Task 4. – Bitwise Operations**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Byte A | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M(Mask) | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| A OR M | x | x | 1 | x | x | x | x | x |
|  |  |  |  |  |  |  |  |  |
| Byte B | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M1(Mask) | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B OR M1 | x | x | x | x | x | x | x | x |
|  |  |  |  |  |  |  |  |  |
| Byte B | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M2(Mask) | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B AND M2 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|  |  |  |  |  |  |  |  |  |
| Byte C | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M2(Mask) | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| C OR M2 | x | 1 | x | x | x | x | 1 | x |
|  |  |  |  |  |  |  |  |  |
| Byte D | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M1(Mask) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D AND M1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
|  |  |  |  |  |  |  |  |  |
| Byte D | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M2(Mask) | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| D OR M2 | x | x | 1 | x | 1 | x | x | 0 |
|  |  |  |  |  |  |  |  |  |
| Byte E | x | x | x | x | x | x | x | x |
| Bit Position | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M(Mask) | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| E XOR M2 | 1 | 1 | 1 | x | x | 1 | 1 | 1 |

**Task 5. Logic circuits and truth tables**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Circuit A** |  |  |  |  |  |  |  |
| **((B.C)NOR/A).B.A)** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| **Circuit B** |  |  |  |  |  |  |  |
| **A.B./C** |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  | A | B | C | B.C | /A | D = (B.C) NOR /A | **Final Output O = D.A.B** |
|  |  |  |  |  |  |  |  |
|  | 0 | 0 | 0 | 0 | 1 | 0 | **0** |
|  | 0 | 0 | 1 | 0 | 1 | 0 | **0** |
|  | 0 | 1 | 0 | 0 | 1 | 0 | **0** |
|  | 0 | 1 | 1 | 1 | 1 | 0 | **0** |
|  | 1 | 0 | 0 | 0 | 0 | 1 | **0** |
|  | 1 | 0 | 1 | 0 | 0 | 1 | **0** |
|  | 1 | 1 | 0 | 0 | 0 | 1 | **1** |
|  | 1 | 1 | 1 | 1 | 0 | 0 | **0** |
|  |  |  |  |  |  |  |  |
|  | A | B | C | /C | **Final Output O = A.B./C** |  |  |
|  |  |  |  |  |  |  |  |
|  | 0 | 0 | 0 | 1 | **0** |  |  |
|  | 0 | 0 | 1 | 0 | **0** |  |  |
|  | 0 | 1 | 0 | 1 | **0** |  |  |
|  | 0 | 1 | 1 | 0 | **0** |  |  |
|  | 1 | 0 | 0 | 1 | **0** |  |  |
|  | 1 | 0 | 1 | 0 | **0** |  |  |
|  | 1 | 1 | 0 | 1 | **1** |  |  |
|  | 1 | 1 | 1 | 0 | **0** |  |  |

Both circuits are equivalent as demonstrated in the truth tables.

**Task 6. Pipelining**

The total time it takes is effectively adding the time it takes to prepare the meal and multiplying by each person. The time for a pipelined process is largest time in each cycle.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Task 6** |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Pipelined** | **Cycle -->** |  |  |  |  |  |
| Robin | 7 | 10 | 15 |  |  |  |
| bryan |  | 7 | 10 | 15 |  |  |
| finchie |  |  | 7 | 10 | 15 |  |
| dan |  |  |  | 7 | 10 | 15 |
|  |  |  | **Time** | **60 minutes** |  |  |
| **Total cycles 6** | **1** | **2** | **3** | **4** | **5** | **6** |
|  |  |  |  |  |  |  |
| **Non pipelined (sequential)** | **Cycle -->** |  |  | Sub-totals |  |  |
| Robin | 7 | 10 | 15 | 32 |  |  |
| bryan | 7 | 10 | 15 | 32 |  |  |
| finchie | 7 | 10 | 15 | 32 |  |  |
| dan | 7 | 10 | 15 | 32 |  |  |
|  |  |  | **Total Time** | **128 minutes** |  |  |
| **Total cycles 16** | **1,5,9,13** | **2,6,10,14** | **3,7,11,15** | **4,8,12,16** |  |  |

Sequentially the resources are not being used while the meal recipe is being read, prepared and cooked. However, pipelining allows the use of resources while another task is being executed effectively improving efficiency and throughput.

Because of the constraint of one recipe the other resources are not used as each cycle each person is at a different stage of execution. Therefore, the extra resources are idle as one instance of each is being used during each stage of the pipeline.

**Task 7 CPU Architecture**

1. Multithreading and multiprocessing
   1. Multiprocessing in terms of hardware has more than one physical processing core to execute multiple processes. They can either be structured in a symmetric way or asymmetric, however asymmetric is based on a different architecture (Tech Differences, 2019). Symmetric multi-processing uses multiple cores (with their own cache), that share a pool of memory through a bus and communicate between each other as they complete processes in the operating system (Tech Differences, 2019; Tutorialspoint.com, 2019). Asymmetric has a master core and slave cores where instructions are delegated from the master core that executes a process in the operating system and controls memory access of each core (Tech Differences, 2019).
   2. Multithreading, however, takes advantage of a pipeline of a processor as one thread is executing another is created while functional units of the processor are idle. This happens with every clock cycle and each thread is executing a different stage per cycle. Multithreading allows a greater throughput efficiently sharing resources and executing multiple threads in a shorter time. A thread is contained in a process (single threaded process), but there can exist multiple threads in a single process (multi-threaded). As an example, a user can therefore have processors (multiprocessing) in the background such as a phone call on loudspeaker while using another application. While in the application the user for instance can be running a word processor on a phone while typing and displaying text at the same time, these are executed using threads (Programmer and Software Interview Questions and Answers, 2019).
2. Circumstances where simultaneous multi-threading can degrade performance
   1. The architecture of the processor may slow down performance due to bottlenecking in some circumstances of an application being executed (En.wikipedia.org, 2019). Bad multithreading can also lead to issues such as deadlocks, starvation, race conditions and Live lock (Kitowska, 2019). These errors are an occurrence due to sharing of resources and data as a result of poor multithreading leading to bottlenecking.

**Task 8. Memory**

<https://scoutapm.com/blog/understanding-page-faults-and-memory-swap-in-outs-when-should-you-worry>

<https://anturis.com/monitors/swap-usage-monitor/>

<https://www.tutorialspoint.com/difference-between-internal-fragmentation-and-external-fragmentation>

On the table which represents memory, not virtual memory, there is a page occupying the space where process A is referencing next by the CPU. Therefore, the process B page 3 will need to be swapped to secondary memory so the process A page can be moved from virtual memory into the frame in memory. The CPU would check if there is an entry in the page table and check if it’s in memory, perform a swap with the file from disk and update the table with the new entry.

Process A referencing the end of page 7 which is not in memory and therefore will generate a page fault (interrupt). This occurs because the page is not in memory and will need to be fetched from virtual memory and mapped to memory. The MMU looks at the table for A processes address space and translates the logical address from virtual memory to the corresponding physical memory. The page is read and transferred from disk storage.

Organising pages close together does not affect RAM speed and information access, but bus size, processor, RAM timings (CAS, RAS to CAS, tRAS, RAS pre-charge) and latencies do. Moreover, RAM is random access and process pages may not execute in an order as other pages may need to be accessed first. Through logical assignment of pages in memory, fragmentation is minimised, however if they were to be located close together, it would increase fragmentation, decrease available memory in page memory management as the frames are a set size in paging.

If Page 7 needs to be brought in from virtual memory, page 3 may or may not get swapped. Algorithms in the operating systems kernel determine what happens to pages in memory, when they get swapped, therefore, because of this page 7 may not replace page 3. Page frames are dynamically allocated to minimise fragmentation and increase compaction, so page 7 will be brought into the appropriate frame for its size. However, due to the algorithm and the use of the page occupying the space will determine if it is replaced.

<https://www.tutorialspoint.com/difference-between-internal-fragmentation-and-external-fragmentation>

<https://www.geeksforgeeks.org/page-replacement-algorithms-in-operating-systems/>

<http://homeworkl.blogspot.com/2009/12/types-of-paging-in-operating-system.html>

<https://answers.microsoft.com/en-us/windows/forum/windows_10-performance/physical-and-virtual-memory-in-windows-10/e36fb5bc-9ac8-49af-951c-e7d39b979938>

<https://en.wikipedia.org/wiki/Translation_lookaside_buffer>

Intel optane memory

<https://www.anandtech.com/show/14249/the-intel-optane-memory-h10-review-two-ssds-in-one>

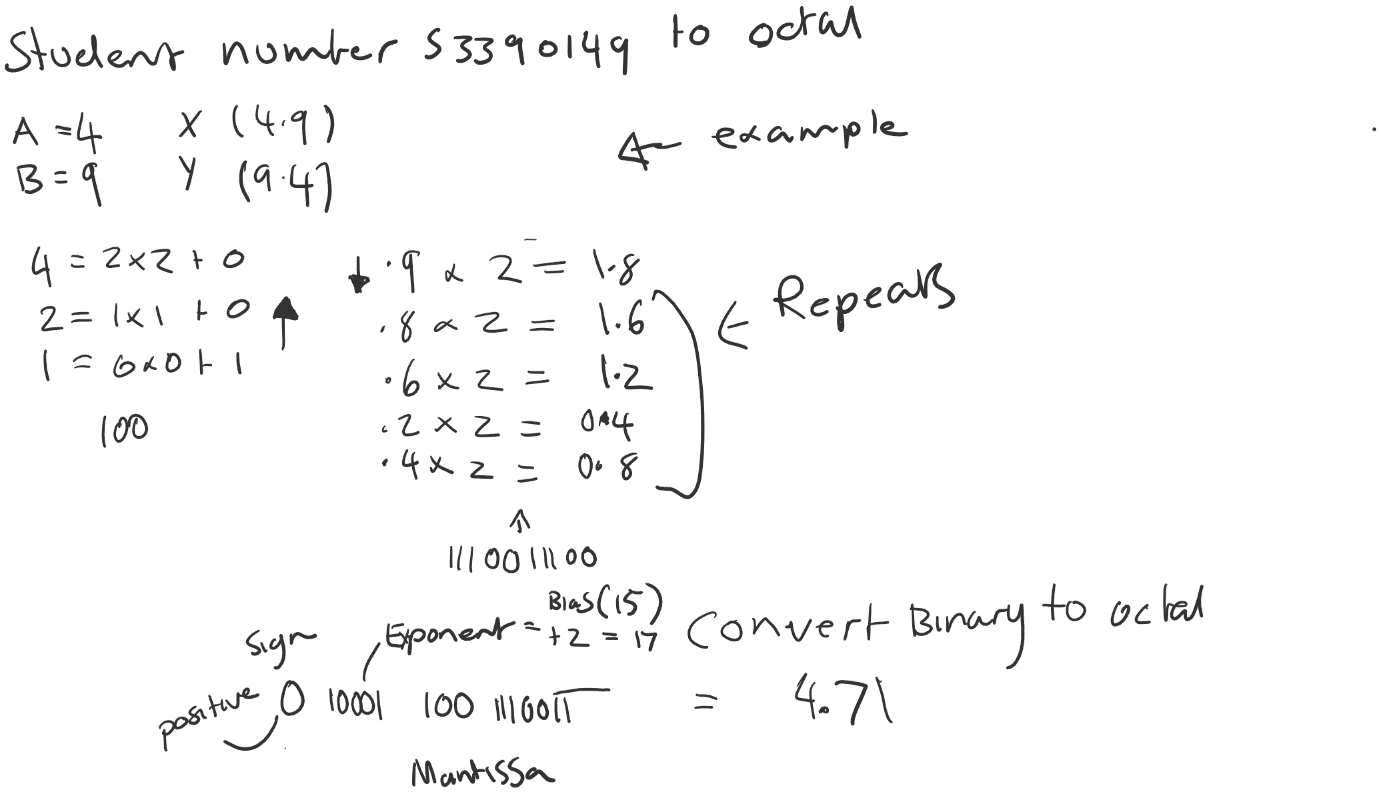
<https://www.anandtech.com/show/14180/pricing-of-intels-optane-dc-persistent-memory-modules-leaks>

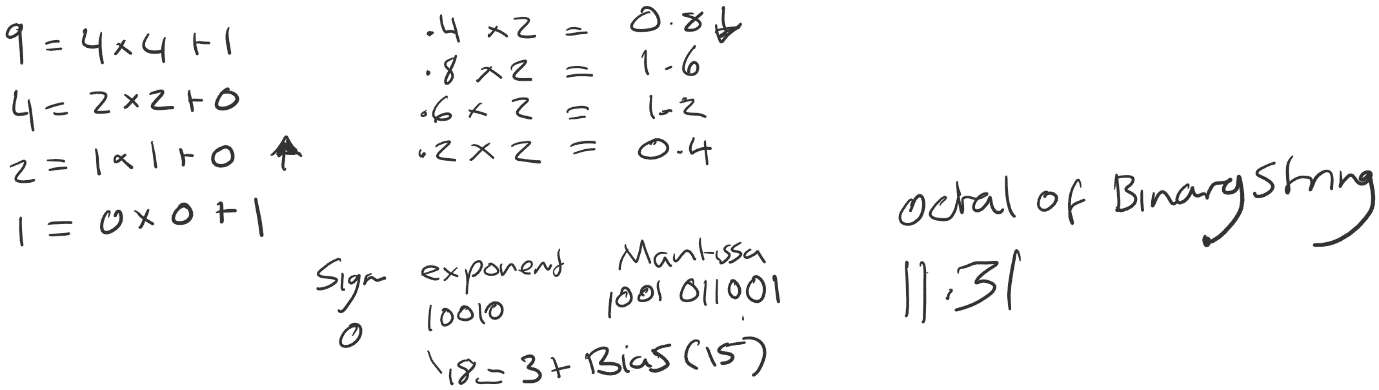
<https://www.tomshardware.com/news/intel-optane-dimm-pricing-performance,39007.html>

<https://www.pcgamer.com/au/intel-optane-memory-everything-you-need-to-know/>

Intel optane is an architecture of memory developed by intel as 3D grid coupled with controllers. It is an architecture designed as separate modules or integrated with storage for use in data centres, cloud, analytics and content delivery. This will only work with a primary drive and not secondary drive where no OS is installed when used with storage as a cache drive. The memory is non-volatile even when power is cut and is in various form factors such as M.2 NVME, integrated with Solid state drives and in Dual inline memory modules.

Designed for data centres with high density 512GB, 256GB and 128GB DIMM modules intel claim are fast, however, they are not as fast as DDR4 RAM. Power consumption is also greater due to the technology and increased capacity. Intel also claim it is also faster and more enduring than NAND flash memory. Optane is also claimed to be fast in all workloads and increase speeds of systems using HDD by using the Optane drive as a cache.

****

****

<https://www.youtube.com/watch?v=mKJiD2ZAlwM> has an example and from this a solution to add x and y from the previous has been implemented below. Depending on the precision of the floating point, calculate which string has the greatest exponent and carry the decimal to line the string with the smallest exponent and add 0 placeholders. Disregard the point and add from right to left following binary addition and include any carry overs from repeated bits not shown relevant to the string and place decimal in the result.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | x | 0 10001 100.1110011 Exponent bias 15 + 2^2 | | | | | | | | | | | | |
|  | y | 0 10010 1001.011001 Exponent bias 15 + 2^3 | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Carry |  |  |  | 1 | 1 | 1 | 1 |  |  | 1 | 1 |  |  |  |
|  |  | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |  |  |  |
|  |  | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  | **1** | **1** | **1** | **0.** | **0** | **1** | **0** | **0** | **1** | **1** |  |  |  |
|  |  | **8** | **4** | **2** |  |  | **2** |  |  | 2 | 1 |  |  |  |
|  | Octal | 16.23 |  |  |  |  |  |  |  |  |  |  |  |  |

Bibliography

Tech Differences. (2019). *Difference Between Symmetric and Asymmetric Multiprocessing (with Comparison Chart) - Tech Differences*. [online] Available at: https://techdifferences.com/difference-between-symmetric-and-asymmetric-multiprocessing.html [Accessed 28 Dec. 2019].

Tutorialspoint.com. (2019). *Symmetric Multiprocessing*. [online] Available at: https://www.tutorialspoint.com/Symmetric-Multiprocessing [Accessed 28 Dec. 2019].

Programmer and Software Interview Questions and Answers. (2019). *Thread vs. Process - Programmer and Software Interview Questions and Answers.* [online] Available at: https://www.programmerinterview.com/operating-systems/thread-vs-process/ [Accessed 28 Dec. 2019].

GeeksforGeeks. (2019). *Difference between Multiprogramming, multitasking, multithreading and multiprocessing - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/difference-between-multitasking-multithreading-and-multiprocessing/ [Accessed 28 Dec. 2019].

En.wikipedia.org. (2019). *Simultaneous multithreading*. [online] Available at: https://en.wikipedia.org/wiki/Simultaneous\_multithreading [Accessed 28 Dec. 2019].

Kitowska, K. (2019). *How To Speed Up Your Device Using Multithreading -*. [online] BoostHigh. Available at: https://boosthigh.com/how-to-speed-up-your-device-using-multithreading/?pk\_campaign=quora&pk\_content=How+does+multithreading+improve+performance?&pk\_kwd=multithreading&pk\_medium=question&pk\_source=quora [Accessed 28 Dec. 2019].