***PART-1***

**PROGRAMMING COMPUTER PROJECT**

**⇒ Constructing Expansion:**

1. **For k = 37:**

initially b=2, q=37, k =0

For 1st iteration, when k = 0:

While (q=37) ≠ 0

A0 = (37 mod 2) = 1

Q = (37 div 2) = 18

**K = k+1 = 1**

For 2nd iteration, when k = 1:

While (q=18) ≠ 0

A1 = (18 mod 2) = 0

Q = (18 div 2) = 9

**K = k+1 = 2**

For 3rd iteration, when k = 2:

While (q=9) ≠ 0

A2 = (9 mod 2) = 1

Q = (9 div 2) = 4

**K = k+1 = 3**

For 4th iteration, when k = 3:

While (q=4) ≠ 0

A3 = (4 mod 2) = 0

Q = (4 div 2) = 2

**K = k+1 = 4**

For 5th iteration, when k = 4:

While (q=2) ≠ 0

A4 = (2 mod 2) = 0

Q = (2 div 2) = 1

**K = k+1 = 5**

For 6th iteration, when k = 5:

While (q=1) ≠ 0

A5 = (1 mod 2) = 1

Q = (1 div 2) = 0

**K = k+1 = 6**

For 7th iteration, when k = 6:

While (q=0) ≠ 0

Now, as loop condition become false. So, loop will break;

Return

A5 A4 A3 A2 A1 A0 = 100101

Output will be:

**(37)10 = (100101)2**

1. **For k = 22:**

initially b=2, q=22, k =0

For the 1st iteration, when k = 0:

While (q=22) ≠ 0

A0 = (22 mod 2) = 0

Q = (22 div 2) = 11

**K = k+1 = 1**

For the 2nd iteration, when k = 1:

While (q=11) ≠ 0

A1 = (11 mod 2) = 1

Q = (11 div 2) = 5

**K = k+1 = 2**

For the 3rd iteration, when k = 2:

While (q=5) ≠ 0

A2 = (5 mod 2) = 1

Q = (5 div 2) = 2

**K = k+1 = 3**

For the 4th iteration, when k = 3:

While (q=2) ≠ 0

A3 = (2 mod 2) = 0

Q = (2 div 2) = 1

**K = k+1 = 4**

For the 5th iteration, when k = 4:

While (q=1) ≠ 0

A4 = (1 mod 2) = 1

Q = (1 div 2) = 0

**K = k+1 = 5**

For the 6th iteration, when k = 5:

While (q=0) ≠ 0

Now, as loop condition become false. So, loop will break;

Return

A4 A3 A2 A1 A0 = 10110

Output will be:

**(22)10 = (10110)2**

**⇒ NOW STEP NO.2**

**(BINARY ADDITION)**

**From Step No.1**

**(37)10 = (100101)2**

**(22)10 = (010110)2**

**Initially, c=0, j=0, n=6**

first iteration, j = 0:

For j=0 to 5

D = (A0=1 + B0=0 + C=0) / 2 = 0

S0 = A0=1 + B0=0 + C=0 – 2(0) = 1

C = 0

**J = 1**

second iteration**, when j = 1:** For j=1 to 5

D = (A1=0 + B1=1 + C=0) / 2 = 0

S1 = A1=0 + B1=1 + C=0 – 2(0) = 1

C = 0

**J = 2**

third iteration**, j = 2:**

For j=2 to 5

D = (A2=1 + B2=1 + C=0) / 2 = 1

S2 = A2=1 + B2=1 + C=0 – 2(1) = 0

C = 1

**J = 3**

fourth iteration, j = 3:

For j=3 to 5

D = (A3=0 + B3=0 + C=1) / 2 = 0

S3 = A3=0 + B3=0 + C=1 – 2(0) = 1

C = 0

**J = 4**

fifth iteration, j = 4:

For j=4 to 5

D = (A4=0 + B4=1 + C=0) / 2 = 0

S4 = A4=0 + B4=1 + C=0 – 2(0) = 1

C = 0

**J = 5**

sixth iteration, j = 5:

For j=5 to 5

D = (A5=1 + B5=0 + C=0) / 2 = 0

S5 = A5=1 + B5=0 + C=0 – 2(0) = 1

C = 0

**J = 6**

7th iteration, **j = 6:**

For loop breaks here.

So, it will return,

**S5S4S3S2S1S0 = 111011**

(100101)2 + (010110)2 = **(111011)2**

**⇒ NOW: STEP NO.3**

**Procedure for multiplication**

Decimal number are: 37 and 22.

From step-1, we get the binary equivalent of these decimal numbers.

(37)10 = (A5 A4 A3 A2 A1 A0)= (100101)2

(22)10 = (B5 B4 B3 B2 B1 B0)= (010110)2

Here, we will multiply these two binary number,

**(100101)2 \* (10110)2 = (?)2**

Initially, **j=0, p=0**

1st iteration, when j = 0:

For j=0 to (n-1) = 5

If((b0=0) = 1) {condition Is false}

Else

**C0 = 0**

2nd iteration, when j = 1:

For j=1 to (n-1) = 5

If((b1=1) = 1)

**C1 = A= 100 1010**

3rd iteration, when j = 2:

For j=2 to (n-1) = 5

If((b2=1) = 1)

**C2 = A= 1001 0100**

4th iteration, when j = 3:

For j=3 to (n-1) = 5

If((b3=0) = 1)

Else

**C3 = 0**

5th iteration, when j = 4:

For j=4 to (n-1) = 5

If((b4=1) = 1)

**C4 = A**

**10 0101 0000**

6th iteration, when j = 5:

For j=5 to (n-1) = 5

If((b5=0) = 1)

Else

**C5 = 0**

So,

C0 C1 C2 C3 C4 C5 = 0 (100 1010) (1001 0100) 0 (10 0101 0000) 0

**So, second loop begin after add these binary numbers**

1st iteration, when j = 0:

For j=0 to (n-1) = 5

P = add (0, C0=0)

= **0**

2nd iteration, when j = 1:

For j=1 to (n-1) = 5

P = add (0, C1=100 1010)

= **100 1010**

3rd iteration, when j = 2:

For j=2 to (n-1) = 5

P = add (100 1010, C2=1001 0100)

= **0 1101 1110**

4th iteration, when j = 3:

For j=3 to (n-1) = 5

P = add (0 1101 1110, C3=0)

= **0 1101 1110**

5th iteration, when j = 4:

For j=4 to (n-1) = 5

P = add (0 1101 1110, C4=10 0101 0000)

= **011 0010 1110**

6th iteration, when j = 5:

For j=5 to (n-1) = 5

P = add (011 0010 1110, C5=0)

= **011 0010 1110**

7th 6th iteration, when j = 6:

Loop breaks.

Output:

(10 0101)2 \* (01 0110)2

= **(011 0010 1110)2**

**Part II: Writing Project**

**2. Look up Bachmann’s original introduction of Big-O notation. Explain how he and others have used this notation.**

Big-O is a mathematical notation that describe the limit of a function when its arguments tend toward a particular value. Big-O classify functions according to their growth, so different functions with same growth can use same notation of big-o. It is also known as **Order of the function** so we use ‘O’ to represent big-o notation.

Few of the major big-o notations are given below:

1. O (1)
2. O (n)
3. O (log n)
4. O (n. log n)
5. O (n2)
6. O (n!)

**Usage:**

1. Bachmann was working on **Analytic number theory**, during this research he introduced Big-o for the first time.
2. Later on, it was adopted by different fields, such as:
3. In **computer science**, big-o is used to categorize different algorithms according to how much time and space requirement grows as size of input grows.
4. In **mathematics**, big-o is used to describe how closely a finite series approximates in a given function, such as **Taylor series.**

**7. Describe the historic trends in how quickly processors can perform operations and use these trends to estimate how quickly processors will be able to perform operations in the next 20 years**

History of processers start from 1978 with and then

1. From **1978 to mid-1980**: First **VAX**-11/780 was introduced, later this series continued with **VAX**-11/785**, VAX**-8700. Then **MIPS** (M/120, M2000) and **IBM** (RS6000) introduced their processes too. During this era growth of processes increased with average of **25%** annually.
2. From **1990-2002**: **Intel** and **Alpha** replaced the old technology with more advance architecture and increased the processing power by average of **52%** annually. This is the era where **first multi-core processer** **VLSI** was introduced by **IBM** in 2001**.** This chip with two **64-bit** microprocessors and have more than **170 million transistors.** With multicore technology a new era started which took the performance of processors to a next level.
3. From 2002 and onward: Intel with 64-bits processers started a new journey and processors now
   1. Core i3
   2. Core i5
   3. Core i7
   4. Core i9

**COMPARISON BETWEEN DIFFERENT PROCESSOPRS**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Date** | **Internal Rag:** | **Clock Speed** | **Data Width** | **Address Lines** | **Max: Memory Space** |
| 8086 | 1974 | 16 Bit | 2 MHZ | 16 bits | 20 Bit | 1MB |
| 80286 | 1982 | 16 Bit | 6 MHZ | 16 bits | 24 Bit | 16 MB |
| 80386 | 1985 | 32 Bit | 16 MHZ | 32 bits | 32 Bit | 4 GB |
| 80486 | 1989 | 32 Bit | 25 MHZ | 32 bits | 32 Bit | 4GB |
| PENTIUM | 1993 | 32 Bit | 60 MHZ | 32 bits, 64 bit bus | 32 Bit | 4GB |
| PENTIUM  II | 1997 | 32 Bit | 233 MHZ | 32 bits. | 32 Bit | 64 GB |
| PENTIUM  III | 1999 | 32 Bit | 450 MHZ | 64 bit bus | 32 Bit | 64 GB |
| PENTIUM  IV | 2000 | 32 Bit | 1.5 MHZ | 32 bits, 64 bit bus | 32 Bit | 64 GB |

Google announced it has a quantum computer that is **100 million times faster** than any classical computer in its lab. Every day, we produce **2.5 exabytes of data**. That number is equivalent to the content on 5 million laptops.

And now in **next 20 years** with technologies like quantam processors and neural networking perfomce of processor will going to to touch a performace **of 10000 million time** faster than today.

This revolution is going to effect:

1. Computation chemistry.
2. Cyber security
3. Weather forecasting
4. Drug Design & Development.
5. Financial Modelling
6. Cryptography and currency.