- \*\*Result\*\*: The sum is \( 00000000 \), which is 0 in decimal.

Thus, \*\*X - Y = 0\*\* (in decimal), and the binary result is \*\*00000000\*\*.

---

### \*\*Question No. 2: What is the purpose of an Instruction Register in a CPU? How does the Instruction Register use Opcode and Operand? How are the number of bits allocated to the Opcode and Operand?\*\*

- \*\*Purpose of the Instruction Register (IR)\*\*:

- The Instruction Register (IR) holds the current instruction that the CPU is executing. It stores the fetched instruction from memory until it is decoded and executed.

- \*\*How the IR uses Opcode and Operand\*\*:

- The \*\*Opcode\*\* is the part of the instruction that specifies the operation to be performed (e.g., ADD, SUB, MOV).

- The \*\*Operand\*\* is the part of the instruction that specifies the data or the memory location on which the operation is to be performed (e.g., register or memory address).

- The Instruction Register is split into two parts: the first part holds the \*\*Opcode\*\*, and the second part holds the \*\*Operand\*\*.

- \*\*How bits are allocated\*\*:

- The number of bits allocated to the \*\*Opcode\*\* and \*\*Operand\*\* depends on the CPU architecture and the instruction format. Typically:

- A fixed number of bits are allocated to the \*\*Opcode\*\* (e.g., 6 bits).

- The remaining bits are allocated to the \*\*Operand\*\* (e.g., 26 bits in some instruction formats like MIPS).

- For example, in a 32-bit instruction format, you might have:

- 6 bits for the Opcode

- 26 bits for the Operand.

---

### \*\*Question No. 3: Consider a hypothetical microprocessor generating a 64-bit address and having a 64-bit data bus. What is the maximum memory address space that the processor can access directly if it is connected to "32-bit memory"?\*\*

- \*\*Address Space\*\*: The microprocessor generates a \*\*64-bit address\*\*, meaning it can address up to \( 2^{64} \) unique locations. This is the \*\*maximum addressable memory\*\* by the processor.

- \*\*32-bit Memory\*\*: If the memory is \*\*32-bit wide\*\*, it means each addressable location holds \*\*32 bits\*\* (4 bytes) of data.

- The maximum memory address space is determined by the number of bits in the address, not by the width of the memory data bus. Since the processor can address \( 2^{64} \) locations, the total memory address space it can access directly is:

\[

2^{64} \text{ locations}.

\]

So, the maximum address space that the processor can access directly is \*\*\( 2^{64} \) locations\*\*. However, the width of the memory (32-bit) means that each address corresponds to 4 bytes of memory, but this does not change the address space.

---

### \*\*Question No. 4: What is the purpose of the Control Register? How are macro instructions and micro instructions related to the control register?\*\*

- \*\*Purpose of the Control Register\*\*:

- The \*\*Control Register\*\* stores control information used by the CPU to manage various operations, such as interrupt handling, memory access, and instruction execution. It helps coordinate the flow of data and operations in the processor.

- \*\*Macro Instructions and Micro Instructions\*\*:

- A \*\*macro instruction\*\* is a high-level instruction, such as "ADD" or "MOV", that the CPU executes as part of a larger program.

- A \*\*micro instruction\*\* is a lower-level instruction that specifies a single step in the execution of a macro instruction. It corresponds to control signals for internal operations (e.g., transferring data between registers, accessing memory).

The \*\*Control Register\*\* is closely related to micro instructions because it holds control signals that direct the CPU's execution of each micro instruction within a macro instruction. When the CPU executes a macro instruction, it translates it into a series of micro instructions, each of which triggers specific control signals stored in the control register.

---

### \*\*Question No. 5: How does the Program Counter point towards the instruction with reference to the System Clock? If the Program Counter is 16-bit wide, what will be the number of locations in the Memory (RAM)?\*\*

- \*\*How the Program Counter (PC) works\*\*:

- The \*\*Program Counter (PC)\*\* holds the address of the next instruction to be executed. It points to the memory location where the next instruction resides.

- With each clock cycle, the \*\*PC\*\* is updated to point to the next instruction. This is typically done by incrementing the \*\*PC\*\* (e.g., PC = PC + 1) after fetching the current instruction. If there are jumps (branch instructions), the \*\*PC\*\* is updated with a new address as specified by the instruction.

- \*\*If the Program Counter is 16-bit wide\*\*, it means the \*\*PC\*\* can hold up to 16 bits, or \( 2^{16} \) different values.

- The number of memory locations that can be addressed by the Program Counter is equal to the maximum number of values the \*\*PC\*\* can hold, which is:

\[

2^{16} = 65,536 \text{ locations}.

\]

Thus, the memory can have \*\*65,536 locations\*\*, or \*\*64 KB of RAM\*\* (if each location holds 1 byte).