Name: Vin Sovichhay

ID: e20241261

Group: I2-AMS2B

TP3

1. What is the functionally of a bus in a CPU and why is it really important for computer operation?

+ Bus : refers to a communication system that connects different components inside a compute, such as the CPU, memory, and input/output devices. It allows these part to exchange data, addresses, and control signals, which helps the computer operate smoothly and use its resources efficiently.
+ Buses are very important in modern computers because they link all the hardware components through an effective communication channel. As mentioned earlier, they allow subsystems like I/O ports, RAM, and ROM to communicate with each other. This system supports fast data transfer and reduces bottlenecks compare to traditional hard-wired connections. Without buses, the different parts of a computer would not be able to work together properly.

2. Explain the differences between the address bus, data bus and control bus in CPU.

Bus type	Main purpose	Direction of Signal	Key difference
Data bus	Transfer the actual information between the CPU, memory, and I/) device.	Bidirectional (data can move to and from the CPU).	Carried the real data or instruction being processed
Address bus	Specifies the memory location or device the CPU wants to access	Unidirectional (from CPU to memory or I/O)	Carries the address or location of the data
Control bus	Sends signals that manage and coordinate operations in the system.	Mostly bidirectional	Carries control commands like read, write, interrupt, and timing signals

3. Research question: how do modern CPUs manage high-speed data transfers efficiency using bus architecture like PCIe or Hyper Transport. (research question).

+ Modern CPUs handle high-speed data transfers efficiently by using advanced bs technologies like Hyper Transport and PCI Express (PCIe). These systems allow fast communication between the CPU and other components by providing higher bandwidth, lower latency, and more organized data flow.
+ Point-to-point communication: Instead of using a shared bus where many devices compete for access, modern architectures use direct links between components. For example, Hyper Transport connects processors and other components directly, while PCIe connects devices through dedicated lanes. This reduces traffic congestion and allows faster data exchange.
+ High bandwidth and scalable links: Both technologies support multiple lanes or links that can carry large amounts of data simultaneously. PCIe can scale using configurations such as x1, x4, x8, or x16 lanes, while Hyper Transport can increase link width and speed. This scalability allows systems to handle demanding workloads efficiently.

+ Low-latency data transfer: These architectures are designed to minimize delays when sending information. Hyper Transport uses short, direct communication paths, while PCIe uses optimized routing through a root complex. This ensures quick responses when the CPU communicates with memory, GPUs, or other devices.
+ Packet-based data transmission: Data is divided into small packets before being sent across the connection. This allows more efficient transmission, better error handling, and improved organization of data traffic.

4. What are the Arithmetic Operations? Please give examples for those operations and tell what operations do.

+ Arithmetic operations are basic mathematic calculations that we can do in programming using arithmetic operators. These operations allow us to work with numbers, calculate values, and manipulate variables in our code. Here are common arithmetic operations, what they do, and examples:

    1. Addition (+): adds two or more numbers together. Example: 5 + 5 gives 10.
    2. Subtraction (-): find the difference between two numbers. Example: 10 – 5 gives 5.
    3. Multiplication (*): multiplies numbers. Example: 5 * 5 gives 25.
    4. Division (/): divides one number by another. Example: 10 / 2 gives 5.
    5. Modulo (%): gives the remainder after dividing two numbers. Example: 10 % 3 gives 1 because 3 goes into 10 three times with 1 left over.
    6. Exponentiation (**): raises a number to the power of another. Example: 2**3 gives 8
    7. Increment (++): increases a number by 1. Example: if a = 9, then a++ makes a = 10.
    8. Decrement (--): decreases a number by 1. Example: if a = 9, then a-- makes a = 8.

5. What are Logic Operations? Please give example for those operations and tell what the operations do.

+ Logic operations are basic operations that work with true of false values (also called Boolean values). They are used in programming and computing to make decisions, control loops, and perform calculations based on conditions. The common logic operations and what they do are:

    1. AND (&& or AND): returns true only if both inputs are true. Example: true AND true = true , true AND false = false.
    2. OR (|| or OR): returns true if at least one input is true. Example: true OR false = true, false OR false = false.
    3. NOT (! or NOT): returns the opposite of the input. Example: NOT true = false, NOT false = true.
    4. XOR (exclusive OR): returns true only If exactly one input is true. Example: true XOR false = true, true XOR true = false.
    5. NAND: returns false only if both inputs are true; otherwise, it returns true. Example: true NAND true = false, true NAND false = true.

6. What is the purpose of Instruction Register (IR), and how does it help in the execution of instructions?

+ Instruction Register (IR) is a part of CPU that holds the instruction currently being executed. Its main job is to tell the CPU what operation to perform. The IR stores the opcode, which shows the specific action like addition, subtraction, or moving data, and it may also hold the operands or addresses needed for that operation.
+ The IR helps in executing instructions through the Fetch-Decode-Execute cycle:

    1. Fetch: The CPU gets an instruction from memory and stores it in the IR.

2. Decode: The IR is used to read the opcode, so the CPU knows what operation to do and what data or addresses are needed.
3. Execute: The CPU carries out the instruction. It might use the ALU for arithmetic or logic operations, or get data from memory. After this, the IR takes the next instruction for processing.

7. What is the memory address register? And what is the function of the memory address register (MAR)?

+ The Memory Address Register (MAR) is a special register inside the CPU that holds the address of a memory location that the CPU wants to access. It does not store the actual data however, it only stores where the data is located in memory so that the CPU can either read from or write to that location.
+ The main function of the MAR is to supply the address on the address bus when the CPU needs to get data or instructions from memory or send data back to memory. During a read operation, the CPU places the address of the required data into the MAR so that memory can send that data back to the CPU. During a write operation, the CPU places the address of where the data should be stored into the MAR before sending the actual data from the Memory Data Register (MDR) to memory.

References

1. https://www.scribd.com/document/804623398/CPU-Buses and https://www.lenovo.com/us/en/glossary/bus/?orgRef=https%253A%252F%252Fwww.google.com%252F&srsltid=AfmBOooXeri0-mwLkO6bchIqEjhLWJxFhXo5_8a2uXwiHO-nfniRJu_O
2. https://en.eeworld.com.cn/news/qrs/eic296757.html
3. https://www.lenovo.com/us/msd/en/glossary/what-is-hypertransport/?orgRef=https%253A%252F%252Fwww.google.com%252F
4. https://www.lenovo.com/us/en/glossary/arithmetic-operator/?orgRef=https%253A%252F%252Fwww.google.com%252F&srsltid=AfmBOorHNu4bNwbTILK5kjKFxf_v20yGZVyuNhayoQdLiF3eNZmQq-xh
5. https://www.lenovo.com/ca/en/glossary/logical-operation/?orgRef=https%253A%252F%252Fwww.google.com%252F&srsltid=AfmBOorl0nqn5pNdP2hXQs5H3k51rs9OWLUZVwUWI9uGQOGZo1x1QC25
6. https://www.vpnunlimited.com/help/cybersecurity/instruction-register?srsltid=AfmBOopAS-3R9Zw4-kR4YxtatE9-s09Tog29izQzOvzxaFI53InRYpWB
7. https://en.wikipedia.org/wiki/Memory_address_register