Past Question Sets Hint Contents:

# Interrupt signals in the 8085 microprocessor

There are 5 interrupt signals in the 8085 microprocessor:

1. **TRAP:** The TRAP interrupt is a non-maskable interrupt that is generated by an external device, such as a power failure or a hardware malfunction. The TRAP interrupt has the highest priority and cannot be disabled.
2. **RST 7.5:**The RST 7.5 interrupt is a maskable interrupt that is generated by a software instruction. It has the second highest priority.
3. **RST 6.5:** The RST 6.5 interrupt is a maskable interrupt that is generated by a software instruction. It has the third highest priority.
4. **RST 5.5:**The RST 5.5 interrupt is a maskable interrupt that is generated by a software instruction. It has the fourth highest priority.
5. **INTR:**The INTR interrupt is a maskable interrupt that is generated by an external device, such as a keyboard or a mouse. It has the lowest priority and can be disabled.

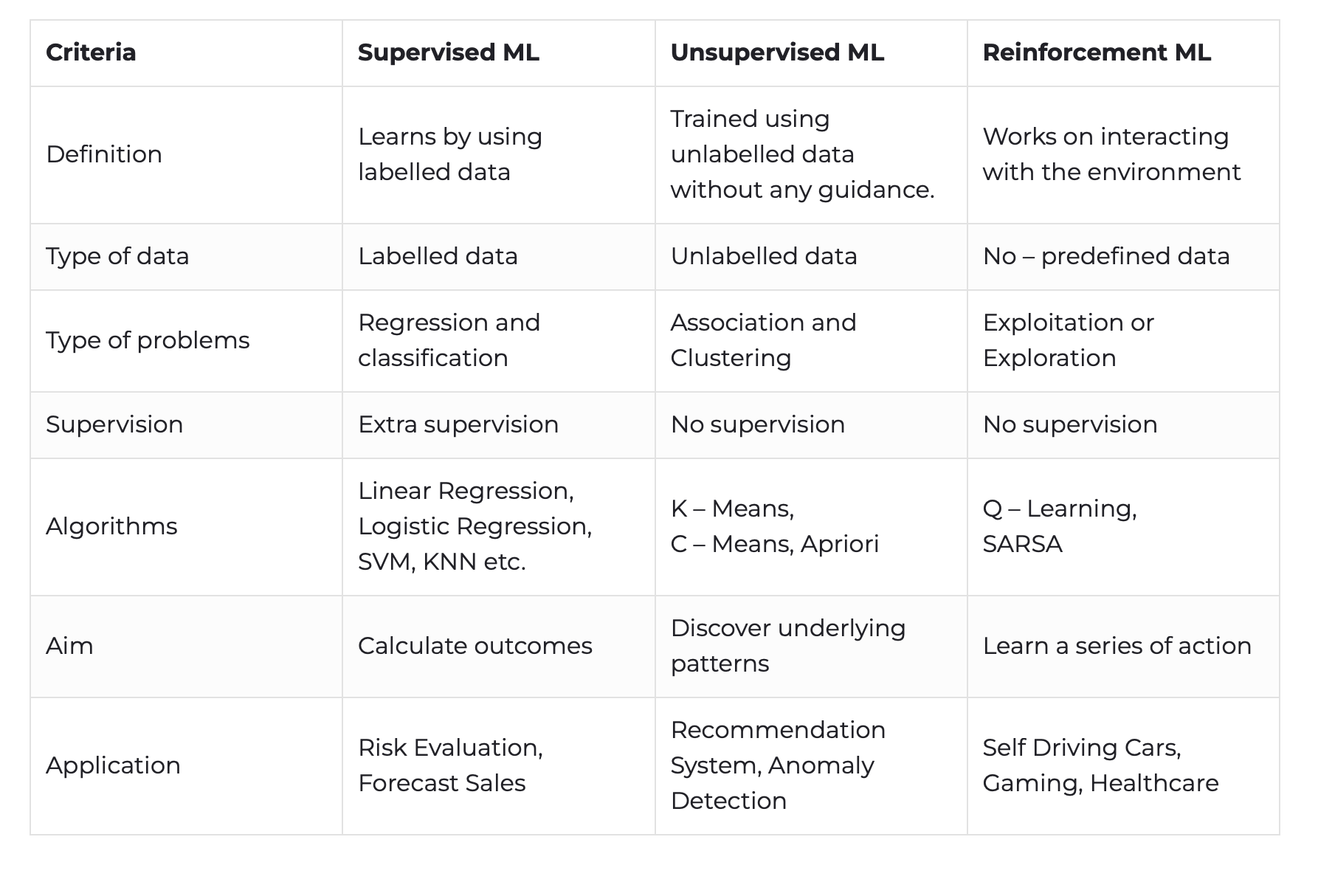
Hardware and Software Interrupts –

When microprocessors receive interrupt signals through pins (hardware) of microprocessor, they are known as *Hardware Interrupts*. There are 5 Hardware Interrupts in 8085 microprocessor. They are – *INTR, RST 7.5, RST 6.5, RST 5.5, TRAP*

*Software Interrupts* are those which are inserted in between the program which means these are mnemonics of microprocessor. There are 8 software interrupts in 8085 microprocessor. They are – *RST 0, RST 1, RST 2, RST 3, RST 4, RST 5, RST 6, RST 7*.

**Vectored and Non-Vectored Interrupts –** *Vectored Interrupts* are those which have fixed vector address (starting address of sub-routine) and after executing these, program control is transferred to that address. Vector Addresses are calculated by the formula 8 \* TYPE

# Supervised, Unsupervised and Reinforcement Learning



# Interrupts in 8086

The different types of interrupts present in the 8086 microprocessor are given by:

1. **Hardware Interrupts –** Hardware interrupts are those interrupts that are caused by any peripheral device by sending a signal through a specified pin to the microprocessor. There are two hardware interrupts in the 8086 microprocessor. They are:
   * *NMI (Non-Maskable Interrupt):* It is a single pin non-maskable hardware interrupt that cannot be disabled. It is the highest priority interrupt in the 8086 microprocessor. After its execution, this interrupt generates a TYPE 2 interrupt. IP is loaded from word location 00008 H, and CS is loaded from the word location 0000A H.
   * *INTR (Interrupt Request):* It provides a single interrupt request and is activated by the I/O port. This interrupt can be masked or delayed. It is a level-triggered interrupt. It can receive any interrupt type, so the value of IP and CS will change on the interrupt type received.
2. **Software Interrupts –** These are instructions inserted within the program to generate interrupts. There are 256 software interrupts in the 8086 microprocessor. The instructions are of the format INT type, where the type ranges from 00 to FF. The starting address ranges from 00000 H to 003FF H. These are 2-byte instructions. IP is loaded from type \* 04 H, and CS is loaded from the following address given by (type \* 04) + 02 H. Some important software interrupts are:
   * *TYPE 0* corresponds to division by zero(0).
   * *TYPE 1* is used for single-step execution for debugging the program.
   * *TYPE 2* represents NMI and is used in power failure conditions.
   * *TYPE 3* represents a break-point interrupt.
   * *TYPE 4* is the overflow interrupt.

# Multiprocessing, Multitasking, MultiThreading, TimeSharing

## 1. Multiprocessing

* **Definition**: Refers to the use of two or more CPUs (or cores) in a computer system to execute processes concurrently.
* **Concurrency**: True parallelism, since multiple processes can literally run simultaneously on different processors or cores.
* **Key Point**: Each processor works independently on a different task. This improves performance as tasks are divided among multiple processors.
* **Example**: Modern multi-core CPUs where different cores execute different processes simultaneously.

## 2. Multitasking

* **Definition**: Refers to the ability of an operating system to manage multiple tasks (or processes) at the same time, giving the illusion that they are running simultaneously.
* **Concurrency**: This is achieved through **context switching**, where the CPU rapidly switches between tasks. Only one task is actually running at any given time on a single CPU, but switching happens so fast that it appears concurrent.
* **Key Point**: Improves CPU utilization by sharing the CPU time among multiple tasks.
* **Example**: Running a browser, word processor, and media player on the same machine, where the OS switches between them quickly.

## 3. Multithreading

* **Definition**: Refers to the ability of a CPU to manage multiple threads within the same process. Threads are smaller units of a process that can be executed concurrently.
* **Concurrency**: In a single-core CPU, threads take turns running, but in multi-core or multi-threaded CPUs, different threads can run simultaneously, offering true parallelism within a process.
* **Key Point**: Threads share the same memory space, making communication between them faster, but they must also manage shared resources carefully to avoid conflicts (e.g., race conditions).
* **Example**: A web browser using one thread for rendering the page and another thread for downloading resources.

## 4. Time-sharing

* **Definition**: A computing technique where multiple users or programs share the same system resources by allocating a specific time slice to each user/process.
* **Concurrency**: This creates the illusion of concurrent execution by rapidly switching between tasks. Like multitasking, time-sharing is achieved through context switching.
* **Key Point**: It focuses on ensuring that all users/programs get fair access to the system resources in a controlled manner.
* **Example**: Early mainframes running multiple user programs on the same CPU, where each user would get a short amount of CPU time in rotation.