Explain how the PREEMPT_RT patches improve the Linux Kernel performance. Examples include interrupt handling, process scheduling, and spinlock synchronization.

Interrupt handling,

- Interrupt handlers run with interrupts disabled
- In PREEMPT_RT, almost all interrupt handlers are threaded
- Very small hardware interrupt handlers are used, that have a well-defined execution time
- They acknowledge the interrupts, and enqueues the “real” interrupt handler
- The interrupt handler runs in a dedicated Kernel thread
- Threaded interrupts are well established in the mainline kernel

Overall, interrupt handling mechanism improves CPU efficiency.

Process scheduling,

The CPU time is shared between applications, the scheduler decides who runs at any given time.

The scheduler is invoked on several occasions:

- When an application waits for external data or events
- When external data or events needs to be processed
- Periodically

When a task runs in user space mode and gets interrupted by an interruption, if the interrupt handler wakes up another task, this task can be scheduled as soon as returned from the interrupt handler.

The scheduler is also a key component in guaranteeing RT behavior.

In general, the scheduler determines when and for how long processes run. Therefore, the scheduler's behavior strongly affects a system's performance.

Spinlock synchronization,

Locks are synchronization primitives that arbitrate concurrent access to a resource.

Spinning locks can be taken with interrupts constraints.

Spinlocks will busy-wait until the lock is freed.

Spinlocks will disable preemption when taken.

So, spinlock will significantly improve performance when resources are sufficient.