I think the I2C problem or issue was caused by an uncontrolled synchronization due to the RTOS implementation. Calling the shared resources multiple times without the proper use of synchronization tools can be a disaster in embedded system.

To fix the issue we need to implement semaphore and mutexes to synchronize and protect the function call. In terms of the initial design whoever calls the i2c\_write function should own the function until it responds. The I2C\_Write needs to be protected by mutex while accessing critical sections. mutex has a priority inheritance in w/c it can prioritized the task who needs to be executed.

each SensorTask needs to take a semaphore to access the shared resources and needs to give semaphore so other task can access the resource.

I think in this way we can prevent data corruption.