

Real-time Mutate Summary

- Scenario:
 - Data is big: `std::atomic<>::is_always_lock_free == false`
 - The real-time thread **can** mutate the object
 - Real-time thread will not fail to acquire the resource
- Trade-off:
 - The non-real-time thread **can not** mutate the object
 - Non-real-time thread will wait on the real-time thread
 - Overhead of copying on the non-real-time & real-time thread
- Examples:
 - Sharing large data from the real-time thread to the non-real-time thread
 - GUI visualisations, frequency spectrums, oscilloscopes etc.

Both Mutating

