5.7 Choosing the Right Tool: A Practical Comparison

Deciding which synchronization mechanism to use is a key skill in embedded programming. Use the following table and questions as a guide.

Mechanism	Purpose	Transfers Data?	When to Use	Primary Risk
Critical Section	Mutual Exclusion	No	Protecting a few lines of code that modify a shared variable. Must be extremely fast.	Halts system responsiveness by disabling all interrupts. Can easily break real-time deadlines.
Semaphore	Signaling	No	Notifying a task that a specific event has occurred. Deferring ISR work to a task.	Does not help with transferring the actual data associated with the event.
Queue	Data Transfer	Yes	Sending data of any type from an ISR to a task for processing.	Minor overhead for copying data into the queue. May not be suitable for very large data structures.

Decision-Making Guide

When designing an interaction between an ISR and a task, ask yourself these questions:

1. **Do I need to pass data from the ISR to the task?**
   - **Yes:** Use a **Queue**. This is the safest and most robust solution for transferring data.
   - **No:** Go to question 2.
2. **Is my goal simply to wake up a task to do some work when an interrupt occurs?**
   - **Yes:** Use a **Semaphore**. It is a lightweight and highly efficient signaling mechanism.
   - **No:** Go to question 3.
3. **Do I only need to protect a single, simple variable (like a counter or flag) during a very quick read-modify-write operation?**
   - **Yes:** A **Critical Section** is an option, but only if the operation is genuinely just a few lines of code. Be aware of the impact on system latency.
   - **No:** Re-evaluate your design. You likely need a semaphore or a queue.

In modern RTOS development, **Queues and Semaphores are almost always preferred over Critical Sections** for managing ISR-task interactions. They provide cleaner, safer, and more scalable solutions that have less impact on the overall real-time performance of your system.

Revision #1
Created 2025-08-28 13:05:13 UTC by GI
Updated 2025-08-28 13:06:07 UTC by GI