Process synchronization is achieved through various synchronization mechanisms and techniques that ensure proper coordination and control of multiple processes or threads, preventing conflicts and data inconsistencies when accessing shared resources.

Here are some common ways process synchronization is achieved:

1. Semaphores: Semaphores are used to control access to shared resources. Processes or threads must acquire a semaphore (through a wait operation) before accessing the shared resource and release it (through a signal operation) after completing the task.
2. Locks (Mutex): Locks, also known as mutual exclusion (mutex) locks, provide exclusive access to a shared resource. Only one process or thread can hold the lock at a time, ensuring mutual exclusion and preventing concurrent access.
3. Condition Variables: Condition variables are used to signal between processes or threads to coordinate their activities. They allow threads to wait for a specific condition to become true before proceeding with their tasks.
4. Monitors: Monitors are high-level synchronization constructs that combine locks and condition variables to simplify process synchronization. They provide a structured approach to access shared resources in a controlled manner.
5. Atomic Operations: Some hardware architectures provide atomic instructions, which ensure certain operations are executed as a single, indivisible unit. Atomic operations help avoid race conditions and data inconsistencies during concurrent access to shared resources.
6. Message Passing: Processes can communicate and synchronize with each other using message passing mechanisms. They can exchange messages to coordinate activities and ensure proper synchronization.
7. Barriers: Barriers are synchronization points where processes or threads must wait until all participating entities reach the barrier before continuing execution. Barriers

are commonly used in parallel computing and parallel algorithms.

8. Read-Write Locks: Read-write locks allow multiple threads to read shared data concurrently but require exclusive access (blocking other readers and writers) for writing data.