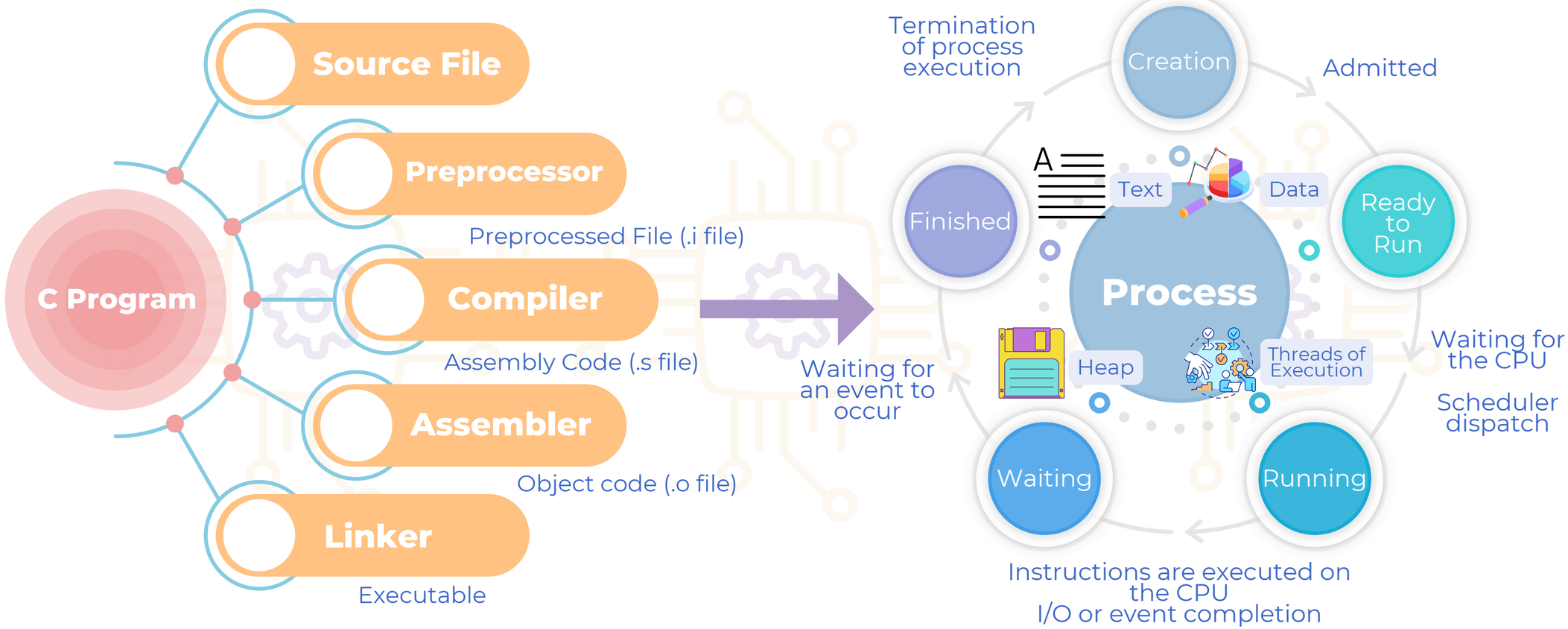
C Program Process Diagram



Virtual Processor

Compilation Process

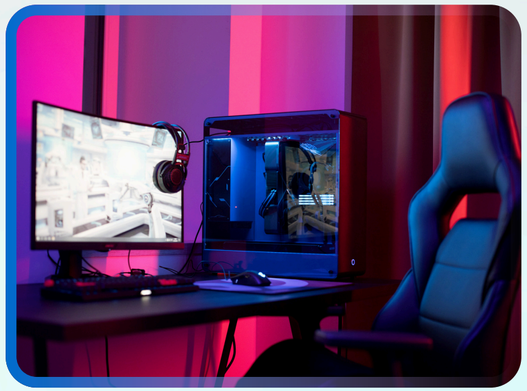
Source File

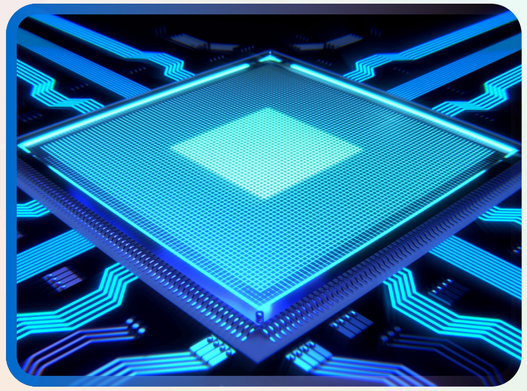
- Definition: The source file is the original code written in the C programming language by the developer. It typically has a .c extension.
- Purpose: This file contains the logic and instructions that the developer wants to execute, serving as the foundation for the entire compilation process.

Preprocessor

- Function: The preprocessor handles all the preprocessing directives like #include, #define, and #if. It replaces macros, includes the contents of header files, and prepares the code for compilation.
- Output: The result is a preprocessed file with a .i extension, which is an expanded version of the source code, ready for the next stage.

Compilation Process





Compiler

- Function: The compiler takes the preprocessed file and translates it into assembly language, a low-level code that is closer to machine language but still human-readable.
- Output: This translation results in an assembly code file with a .s extension, which outlines the program's operations in terms of the target CPU's instruction set.

Compilation Process

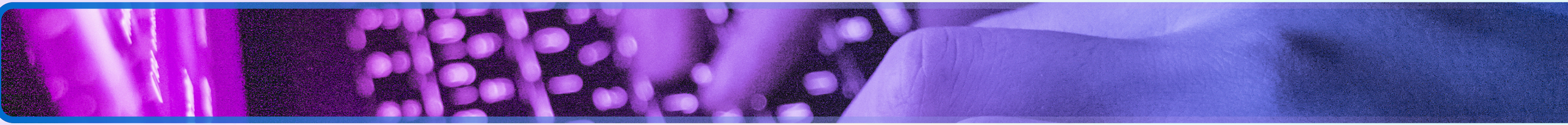
Assembler

- Function: The assembler converts the assembly code into object code, which is a binary representation of the program's instructions, specific to the machine's architecture.
- Output: The output is an object file with a .o extension, containing the machine code but not yet linked into a complete executable.

Linker

- Function: The linker combines all the object files, along with any necessary libraries, into a single executable file. It resolves references between different modules and ensures everything is in place for execution.
- Output: The final product is an executable file, typically with no extension or a .exe on Windows, which can be directly executed by the operating system.

Process Lifecycle



Creation

This is the initial stage where the operating system allocates resources and loads the executable into memory. The process is created but not yet ready to run.

Ready to Run

At this stage, the process is ready to be executed but is waiting for the CPU to become available. It's in a queue, managed by the operating system, waiting for its turn to be dispatched.

Running

The process is actively being executed on the CPU. During this stage, the process performs its intended tasks, such as computations or handling user inputs.

Process Lifecycle

Waiting
If the process needs to wait for an external event (like I/O operations) or for a specific condition to be met, it enters the Waiting state. Here, it's paused and will not consume CPU resources until the event occurs.

Finished
Once the process has completed all its tasks, it enters the Finished state. The operating system then deallocates the resources used by the process, and the process is terminated.