

1. Program Counter (PC):

The process begins with the PC, which holds the address of the next instruction to be executed. It keeps updating automatically as the program runs.

2. Instruction Fetch:

The instruction at the address in the PC is fetched from memory. This is handled by the fetch unit.

3. Instruction Decode:

The fetched instruction is then sent to the decode unit. It identifies what kind of instruction it is (e.g., arithmetic, logical, load/store) and prepares the control signals.

4. Register File Access:

The operands needed for the instruction are fetched from the register file. These are the general-purpose registers like r0 to r15.

5. Barrel Shifter:

If the instruction needs shifting or rotation (which is common in ARM), the operand goes through the barrel shifter. This allows combining shift and operation in a single instruction.

6. ALU (Arithmetic Logic Unit):

The actual computation is done here. It performs operations like addition, subtraction, bitwise logic, etc.

7. Memory Access:

For instructions like LDR or STR, memory access is required. The data is either read from memory or written to it, depending on the instruction.

8. Write-back:

The final result of the operation is written back into a destination register so it can be used later.

9. Flag Update (CPSR):

Depending on the instruction, condition flags like N (Negative), Z (Zero), C (Carry), and V (Overflow) are updated in the Current Program Status Register (CPSR).

This flow is repeated for every instruction and is highly optimized in ARM to allow fast execution using pipelining, where multiple instructions can be processed at different stages simultaneously.