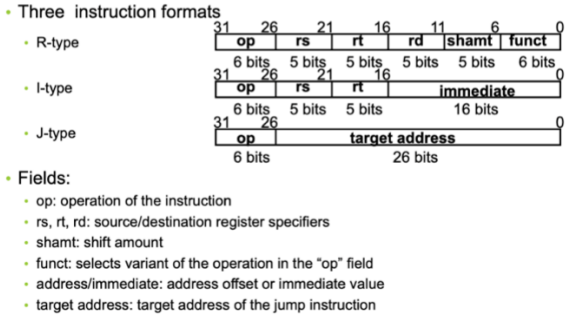
Instruction Execution

MIPs uses a streamlined instruction set architecture with 3 main types of instruction formats: R-type, I-type, and U-type. These formats define how bits are aranged in a 32-bit instruction. Each has a specific layout depending on the operation.



- R-type: used for register based instructions like arithmetic and logical functions.
- I-type: used when immediate value or offset is needed
- J-type: used for jumps

The basic instruction fields include op, which identifies the operation; rs, rt, and rd, which are source and destination register specifiers. Shampt is the shift amount used in shift instructions and funct further defines the specific operation within the op category. The immediate field is for constants or memory offsets, and the target address determines where control should jump to in code.

The rs and rt fields of an I-type instruction represent source registers, but their roles vary based on the instruction. The rt field sometimes acts as a destination (addi, lw) and other times as a source (sw, beq). For example, addi adds an immediate value to rs and stores the result in rt. In contrast, sw stores the value from rt into memory using the address computed from rs+offset.

The MIPs subset includes a variety of operations categorized into:
- Arithmetic/Logical: Instructions such as add, sub, and, or, and slt are employed for basic arithmetic and logic operations.
- Memory references: Instructions like lw and sw are essential for moving data between registers and memory.
- Control transfer: Operations such as beq and jare essential for altering the flow of executions within programs

Instruction Execution:
1. Fetch: The program counter (PC) directs the fetching of instructions from memory. This initiates the sequence of instruction execution
2. Decode Stage: The opcode is analyzed to determine the instruction type - arithmetical, logical, memory, or branch. This stage configures subsequent