

Assessment Questions

I. Describe the difference between RISC and CISC architectures.

RISC	CISC
RISC is a reduced instruction set.	CISC is a complex instruction set.
The number of instructions is less as compared to CISC.	The number of instructions is more as compared to RISC.
The addressing modes are less.	The addressing modes are more.
It works in a fixed instruction format.	It works in a variable instruction format.
The RISC consumes low power.	The CISC consumes high power.
The RISC processors are highly pipelined.	The CISC processors are less pipelined.
Requires more RAM.	Requires less RAM.

II. Explain the role of the program counter (PC) in ARM architecture.

The Program Counter (PC) is a special-purpose register that holds the address of the next instruction to be fetched and executed. It updates automatically after each instruction, ensuring the processor knows the next instruction in the sequence. During branching, the PC is modified to point to a different memory location, enabling non-sequential code execution. In ARM, the PC can also be accessed for program flow control. It plays a central role in ensuring smooth execution and efficient program navigation.

III. What is the significance of condition codes in ARM assembly?

Condition codes in ARM assembly include flags like Zero, Negative, Carry, and Overflow, which are updated after arithmetic or logical operations. They provide status information about the result of the last operation. These codes are essential for conditional execution, allowing instructions to execute only when specific conditions are met. This reduces the need for frequent branching and improves code efficiency. By leveraging condition codes, ARM enables compact and optimized control flow in programs.

IV. How does the ARM pipeline improve performance?

ARM uses pipelining to increase instruction throughput by overlapping different stages of instruction execution, such as fetch, decode, and execute. This means while one instruction is being executed, the next is being decoded, and a third is being fetched. Pipelining reduces the time required to execute a series of instructions, improving overall performance. ARM processors often use 3-stage, 5-stage, or more advanced pipelines. This architecture maximizes the use of resources and minimizes idle time.

V. Compare direct and indirect addressing modes in ARM assembly.

In direct addressing, the memory address of the operand is explicitly specified in the instruction. This makes it simple to use but limits flexibility since the address is fixed. Indirect addressing, on the other hand, uses a register to hold the memory address, allowing the address to be dynamically changed during program execution. Direct addressing is faster for accessing static data, while indirect addressing is more versatile for dynamic data structures like arrays. ARM supports both modes, enabling flexible programming.

VI. What is the difference between LDR and LDM instructions?

The LDR instruction is used to load a single value from a specified memory address into a register. It is simple and efficient for accessing individual memory locations. The LDM instruction (Load Multiple) allows loading multiple values into several registers from a block of contiguous memory addresses in one instruction. LDM is useful for quickly restoring register states or loading array elements. Both instructions improve memory access efficiency, with LDM reducing overhead in multi-register operations.

VII. Explain the use of the stack pointer (SP) in subroutine calls.

The Stack Pointer (SP) is a special register that points to the top of the stack in memory. It is used to manage dynamic memory during function calls, storing local variables, return addresses, and saved registers. When a subroutine is called, the SP adjusts to allocate space for the subroutine's context. Upon returning, the SP restores the previous state of the program. The SP ensures that subroutine calls are managed efficiently and supports nested function calls.

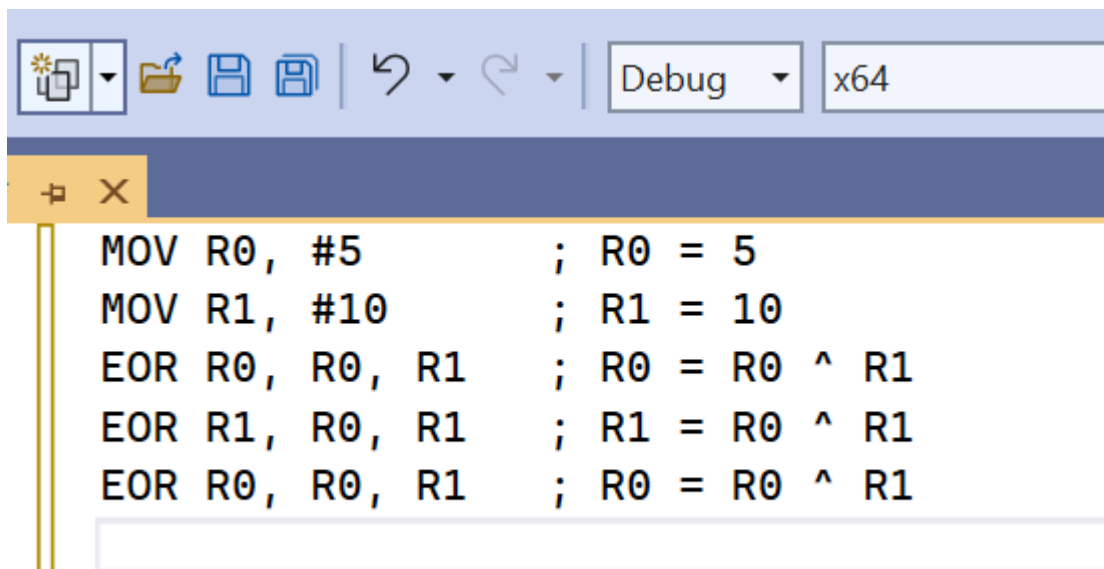
VIII. How are interrupts handled in ARM architecture?

Interrupts in ARM are handled by switching to specific modes like IRQ (Interrupt Request) or FIQ (Fast Interrupt Request). The processor saves the current Program Counter (PC) and CPSR (Current Program Status Register) to ensure program continuity. It then jumps to an interrupt vector table to execute the appropriate interrupt handler. Once the interrupt is processed, the processor restores the saved state and resumes execution. This mechanism ensures prompt handling of critical events without disrupting the main program.

IX. What are the advantages of using thumb instructions in ARM?

Thumb instructions in ARM are 16-bit instructions that reduce code size compared to 32-bit ARM instructions. This compact code improves memory efficiency, making it ideal for embedded systems with limited storage. Despite their smaller size, Thumb instructions maintain performance close to standard ARM instructions. Switching between ARM and Thumb modes is seamless, allowing developers to balance code size and execution speed. This feature enhances ARM's versatility in resource-constrained environments.

X. Write a code snippet for swapping two numbers without using a third variable.



```
MOV R0, #5          ; R0 = 5
MOV R1, #10         ; R1 = 10
EOR R0, R0, R1       ; R0 = R0 ^ R1
EOR R1, R0, R1       ; R1 = R0 ^ R1
EOR R0, R0, R1       ; R0 = R0 ^ R1
```

XI. Define the term "endianess" and its impact on memory storage in ARM.

Endianess refers to the order in which bytes are stored in memory for multi-byte data types. In little-endian systems, the least significant byte (LSB) is stored first, while in big-endian systems, the most significant byte (MSB) is stored first. ARM processors can operate in either endian mode. Endianess impacts how data is interpreted in memory, particularly when transferring data between systems with different endian formats, requiring careful handling to avoid errors.

XII. How does the barrel shifter in ARM instructions work?

The barrel shifter in ARM allows efficient data manipulation by performing shifts and rotations as part of another instruction. For example, it can shift a value left or right or rotate bits within a single instruction cycle. This eliminates the need for separate shift instructions, saving clock cycles and enhancing performance. The barrel shifter is particularly useful in arithmetic operations, bit manipulation, and optimizing complex calculations.

XIII. Why is pipelining important in ARM processors?

Pipelining is crucial in ARM processors as it enables multiple instructions to be processed simultaneously by overlapping their fetch, decode, and execute stages. This reduces the overall time required to execute instructions and increases throughput. Pipelining minimizes idle time for the processor and ensures efficient use of hardware resources. It also improves performance without increasing clock speed, making it a key feature in modern processor design.

XIV. Explain how floating-point operations differ from integer operations.

Floating-point operations handle real numbers with decimal points, requiring specialized floating-point units (FPUs) for precise calculations. These operations are more complex and slower than integer operations, which deal with whole numbers. Floating-point arithmetic is essential in applications like scientific computations and graphics. Integer operations, on the other hand, are simpler and faster, making them suitable for general-purpose tasks like counting and basic arithmetic.

XV. What are the advantages of inline assembly in ARM-based C programming?

Inline assembly allows developers to embed assembly code directly within C programs, combining the efficiency of low-level instructions with the flexibility of high-level languages. It provides direct hardware access, enabling performance optimization for critical sections. Inline assembly is particularly useful for tasks like manipulating registers, implementing custom instructions, or optimizing algorithms. It also allows seamless integration with C variables, improving code readability and maintainability.

Created By: Husnain Ali