Assignment\_08-08-2024

Author: ThanhTH10

Date: 09/08/2024

1. What is opcode and operands.

**Opcode:**

- The opcode (operation code) is the part of a machine instruction that specifies the operation to be performed.

- It tells the processor what kind of operation it should execute, such as adding two numbers, loading a value from memory, or branching to a different part of the program.

- The opcode is typically represented by a binary code or a hexadecimal value.

- For example, in the instruction "ADD R1, R2, R3", the opcode would be the binary or hexadecimal code that represents the "ADD" operation.

**Operands:**

- Operands are the data or values that the operation specified by the opcode should be applied to.

- They are the inputs or parameters required by the operation.

- Operands can be registers, memory addresses, immediate values, or a combination of these.

- In the example "ADD R1, R2, R3", the operands are the register names R1, R2, and R3, which specify the source and destination operands for the addition operation.

1. What is ARM processor. Addressing modes of ARMV8 architecture

**ARM** (Advanced RISC Machine) is a family of reduced instruction set computer (RISC) architectures widely used in embedded systems and mobile devices due to their power efficiency and performance.

**Addressing Modes in ARMv8 Architecture**

1. Immediate Addressing:

- The operand is a constant value specified in the instruction.

- Example: MOV X0, #42

1. Register Addressing:

- The operand comes from a register.

- Example: ADD X1, X2, X3

1. Register with Offset Addressing:

- The operand is a value in a register plus an immediate offset.

- Example: LDR X0, [X1, #8]

1. Register with Register Offset Addressing:

- The operand is a value in a register plus another register’s value.

- Example: LDR X0, [X1, X2, LSL #3]

1. Pre-indexed Addressing:

- The address is calculated with an offset added to the base register, updating the base register.

- Example: LDR X0, [X1, #8]!

1. Post-indexed Addressing:

- The address uses the base register, then adds an offset after the operation.

- Example: LDR X0, [X1], #8

1. PC-relative Addressing:

- The address is calculated by adding an offset to the current Program Counter (PC).

- Example: LDR X0, [PC, #8]

1. Difference between RISC and CISC

| **Characteristic** | **RISC** | **CISC** |
| --- | --- | --- |
| Instruction Set Complexity | Smaller, simpler, and more uniform instruction set | Larger, more complex instruction set |
| Instruction Execution | Load/store architecture (separate load and store instructions) | Can directly operate on memory (complex instructions) |
| Pipelining | Deeper instruction pipelines, allowing for higher clock speeds and better parallelism | Shorter or more complex pipelines |
| Chip Complexity | Simpler chip designs, fewer transistors, more efficient and easier to manufacture | More complex chip designs, more transistors and control logic |
| Performance Optimization | More optimized for performance (e.g., branch prediction, speculative execution, out-of-order execution) | Rely more on the complexity of instructions to achieve performance |
| Programming and Compiler Support | Easier to program and have better compiler support | Require more complex compilers and programming techniques to effectively utilize the instruction set |

1. Constructor A(int x, int y):x(x),y(y){} // What modification needed

This constructor already perfect, but in some case, to avoid the conflict of variable name, we can using:

1. Use a different name for the parameter: As you mentioned, one way is to use a different name for the parameter, such as a and b instead of x and y. This approach is often used when the parameter names are not important or can be easily changed.

class A

{

    int x, y;

public:

    A(int a, int b) : x(a), y(b) {}

};

1. Use the this pointer: Another way is to use the this pointer to explicitly refer to the member variables. This approach is often used when the parameter names are important or cannot be changed.

class A

{

    int x, y;

public:

    A(int x, int y)

    {

        this->x = x;

        this->y = y;

    }

};