Nang Thiri Wutyi

20113

6/14/2024

**Homework Assignment 2**

1. **Discuss how stack architecture computer works by giving examples, such as arithmetic express in reverse polish notation. And compare the pros and cons between stack-based virtual machine and register-based virtual machine.**

The computers which use Stack-based CPU Organization are based on a data structure calleda **stack**. The stack is a list of data words. It uses the **Last In First Out (LIFO)** access method which is the most popular access method in most of the CPU. A register is used to store the address of the topmost element of the stack which is known as **Stack pointer (SP)**. In this organization, ALU operations are performed on stack data. It means both the operands are always required on the stack. After manipulation, the result is placed in the stack.

The main two operations that are performed on the operators of the stack are **Push** and **Pop**. These two operations are performed from one end only.

**1. Push –**   
This operation results in inserting one operand at the top of the stack and it increases the stack pointer register.

It inserts the data word at a specified address to the top of the stack.

**2. Pop –**

This operation results in deleting one operand from the top of the stack and decreasing the stack pointer register.It deletes the data word at the top of the stack to the specified address.

Operation type instruction does not need the address field in this CPU organization. This is because the operation is performed on the two operands that are on the top of the stack. This instruction contains the opcode only with no address field. It pops the two top data from the stack, subtracting the data, and pushing the result into the stack at the top.

Eg. Input: ["10", "6", "9", "3", "+", "-11", "\*", "/", "\*", "17", "+", "5", "+"]

Output: 22

Explanation:

((10 \* (6 / ((9 + 3) \* -11))) + 17) + 5

= ((10 \* (6 / (12 \* -11))) + 17) + 5

= ((10 \* (6 / -132)) + 17) + 5

= ((10 \* 0) + 17) + 5

= (0 + 17) + 5

= 17 + 5 = 22

**The pros of Stack-based CPU organization –**

* Efficient computation of complex arithmetic expressions.
* Execution of instructions is fast because operand data are stored in consecutive memory locations.
* The length of instruction is short as they do not have an address field.

**The cons of Stack-based CPU organization –**

* The size of the program increases.

**The pros of Register-Based Virtual Machines**

* **Performance**: Faster execution due to direct operand access, reducing stack operation overhead.
* **Optimizations**: More opportunities for advanced optimizations like register allocation and instruction reordering. Reduced Stack Management simplifies optimizations and reduces instructions needed for stack manipulation.
* **Parallelism**: Instruction-Level Parallelism makes it asier to exploit due to explicit nature of register-based instructions.
* **Flexible Instruction Set**: Supports a richer set of instructions, allowing for more complex operations.

### **Cons of Register-Based Virtual Machines**

* **Complexity**
* **Compiler Complexity**
* **Larger Bytecode**
* **Fixed Register Limit**:

1. **Processors are one of the most important components in computing systems. Its performance can have a big impact on the whole system. Discuss about processor design metrics and benchmarking tools.**

The evaluation of processor performance involves identifying and analyzing various metrics to understand and improve the overall efficiency and speed of computing systems. As technology advances, new architectures emerge, requiring continual assessment to pinpoint bottlenecks and innovate better designs. Key performance metrics include clock speed, instructions per cycle (IPC), power consumption (measured through metrics like Thermal Design Power, or TDP), and energy efficiency, often represented as performance per watt. Additionally, factors such as die size, transistor count, latency, and throughput play crucial roles in determining a processor's capabilities. Scalability, supported by multi-core and multi-threading technologies, is vital for handling increasing workloads. Architectural features like pipelining, superscalar execution, and out-of-order execution also impact performance, alongside the cache and memory hierarchy.

Benchmarking tools like SPEC, PassMark, Geekbench, Cinebench, AIDA64, 3DMark, Linpack, and Prime95 provide standardized methods to measure and compare these performance metrics across different processors. These tools assess various aspects such as compute-intensive tasks, power efficiency, real-world application performance, and stress testing. Understanding performance metrics is crucial for both consumers and designers. From a purchasing perspective, it helps in making informed decisions, cutting through marketing hype, and ensuring judicious choices by comparing performance, cost, and cost-per-performance ratios. From a designer's perspective, it aids in evaluating design options to achieve the best performance improvements with minimal costs.

The core of performance evaluation often revolves around time, primarily execution time (response time) and throughput. Execution time, the period from task initiation to completion, is critical for individual users, while throughput, the total amount of work done in a given time, is essential for data center managers. Performance is inversely related to execution time; thus, minimizing execution time maximizes performance. For example, if computer A runs a program in 10 seconds and computer B takes 20 seconds for the same task, A is twice as fast as B. Execution time can be calculated by multiplying the total number of clock cycles by the clock cycle time or dividing the total clock cycles by the clock rate. To enhance performance, either the clock cycle time must be reduced or the number of clock cycles per program minimized.

Several factors influence the three key components of CPU execution time: instruction count, clock cycle time, and cycles per instruction (CPI). Instruction count depends on the algorithm, programmer skill, compiler efficiency, and ISA. Clock cycle time is affected by the CPU's organizational structure (e.g., pipelined vs. non-pipelined) and technology used. CPI varies with the program, compiler, ISA, and CPU organization. Designers must optimize these factors to reduce execution time and improve performance. This involves understanding how different instructions and their implementation, memory hierarchy, and I/O handling contribute to performance. For instance, improving cache efficiency or optimizing instruction execution can significantly impact overall performance.

In conclusion, evaluating processor performance through detailed metrics and standardized benchmarks is essential for both designers and consumers. It enables the identification of areas for improvement, guiding the development of more efficient and powerful processors. By understanding the CPU performance equation and the factors affecting it, stakeholders can make informed decisions to enhance system performance, ensuring that the advancements in technology translate into tangible benefits in computing power and efficiency.

Benchmarking tools are essential for assessing and comparing the performance of processors. They provide standardized methods to evaluate various aspects of CPU performance, ensuring that users, developers, and researchers can make informed decisions based on reliable data. Key tools include SPEC, PassMark, Geekbench, Cinebench, AIDA64, 3DMark, Linpack, and Prime95.

* **SPEC CPU** evaluates compute-intensive tasks, while **SPECpower** focuses on power efficiency.
* **PassMark** measures various CPU functions like integer and floating-point operations.
* **Geekbench** offers cross-platform benchmarks for single and multi-core performance.
* **Cinebench** assesses multi-core rendering capabilities.
* **AIDA64** provides detailed system information and benchmarks for CPU and memory.
* **3DMark** evaluates gaming performance, including CPU stress tests.
* **Linpack** measures floating-point performance, especially in HPC contexts.
* **Prime95** is used for CPU stress testing and stability checks.

These tools are essential for standardizing performance evaluations, optimizing processor designs, making informed purchasing decisions, and driving competition and innovation. By providing insights into various performance aspects such as single-core efficiency, multi-core scalability, power consumption, and system stability, benchmarking tools help identify strengths and weaknesses, guiding improvements in processor technology.