San Francisco Bay University

EE488 - Computer Architecture
Homework Assignment #2
Due day: 6/16/2024

Instruction:

1. Push the answer sheet to Github in word file
2. Overdue homework submission could not be accepted.
3. Takes academic honesty and integrity seriously (Zero Tolerance of Cheating & Plagiarism)

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 (1.5~2 pages)

Answer:

A computer with a stack architecture computes on a stack. A linear data structure that follows the Last-In, First-Out (LIFO) principle is the stack. This indicates that the initial thing to be removed is the one that was added most recently. All operations in a stack architecture are carried out via the stack, with instructions subconsciously utilizing the top elements of the stack.

Components of Stack Architecture

Stack: A memory area where data is stored temporarily.
Stack Pointer (SP): A register that points to the top of the stack.
Instruction Set: A set of instructions that operate on the stack.

Workings of Stack Architecture

Basic Operations
Push: Adds an item to the top of the stack.
Pop: Removes the item from the top of the stack.
Top: Refers to the current top element of the stack without removing it.

Instruction Execution

Instructions in a stack architecture typically do not have operands because they implicitly operate on the top elements of the stack. Here's how common instructions work:

PUSH X: Push the value X onto the stack.
POP: Remove the top value from the stack.
ADD: Pop the top two values, add them, and push the result back onto the stack.
SUB: Pop the top two values, subtract the second popped value from the first, and push the result.
MUL: Pop the top two values, multiply them, and push the result.
DIV: Pop the top two values, divide the first by the second, and push the result.

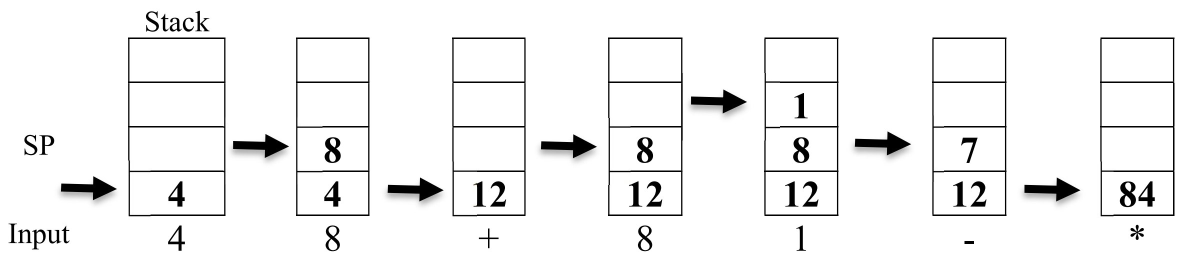
Example: Arithmetic Expression in Reverse Polish Notation (RPN)

RPN, also known as postfix notation, is a mathematical notation in which every operator follows all its operands. It does not need any parentheses if the operators have a fixed number of operands. This makes it straightforward to evaluate using a stack.

Example Expression: (4 + 8) * (8 - 1)

In RPN, this expression is written as:

RPN = 4 8 + 8 1 - *



Steps

1. Initial Stack (empty):

   Stack: [ ]

2. Push 4:

   Stack: [4]

3. Push 8:

   Stack: [4, 8]

4. Add (4 + 8):
   - Pop 8 and 4 from the stack.
   - Add: 4 + 8 = 12.
   - Push the result (12), back onto the stack.

   Stack: [12]

5. Push 8:

   Stack: [12, 8]

6. Push 1:

   Stack: [12, 8, 1]

7. Subtract (8 - 1):
   - Pop 1 and 8 from the stack.
   - Subtract: 8 - 1 = 7.
   - Push the result (7), back onto the stack.

   Stack: [12, 7]

8. Multiply (12 * 7):
   - Pop 7 and 12 from the stack.
   - Multiply: 12 * 7 = 84.
   - Push the result (84), back onto the stack.

   Stack: [84]

The final result of the expression is 84, which is the value remaining in the stack.

Pros and Cons of Stack-Based Virtual Machine

Pros:

a. Simplicity:

   Implementation: Easier to implement due to the straightforward nature of stack operations.
   Instruction Set: Instructions are compact as they do not need to specify operands explicitly.

b. Memory Efficiency:

   Code Size: Smaller code size because instructions are more compact.
   Data Locality: Data is managed efficiently on the stack, reducing the need for memory access.

c. Orthogonality:
All operations implicitly use the stack, which simplifies the design of the instruction set.

Cons:

a. Performance:
Speed: Generally slower than register-based machines due to the overhead of frequent memory accesses.
Stack Operations: Each operation requires pushing and popping from the stack, which can be less efficient than using registers.

b. Limited Parallelism:
Harder to exploit instruction-level parallelism because instructions operate sequentially on the stack.

c. Debugging and Maintenance:
Debugging can be harder due to the implicit nature of operand locations.

Pros and Cons of Register-Based Virtual Machine

Pros:

a. Performance:
Speed: Faster execution as registers provide quicker access to operands than memory.
Optimization: Easier to optimize and exploit instruction-level parallelism.

b. Parallelism:
Facilitates more advanced compiler optimizations and parallel execution.

c. Flexibility:
More flexible instruction set, as registers can hold intermediate values, allowing more complex operations.

Cons:

a. Complexity:
Implementation: More complex to implement due to the need to manage a register file.
Resource Management: Requires careful handling of register allocation and spilling.

b. Instruction Size:
Instructions can be larger because they need to specify the registers to use for each operation, leading to larger bytecode size.

2. 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 (1.5~2 pages)

Answer:

Processor design metrics and benchmarking tools are essential for understanding and evaluating the performance of processors. According to the uploaded documents, here are the key aspects to consider:

Processor Design Metrics:

a. Frequency (Clock Speed)
Definition: The speed, represented by Hertz (Hz), that occurs when a computer's CPU executes instructions. The unit of measurement utilized by contemporary processors is the gigahertz (GHz).

Impact: more rapid times for processing are frequently attained by increasing clock speeds, but these improvements develop at the expense of increased energy consumption and generation of heat.

b. Instructions per cycle (IPC)

Definition: the set of instructions which the computer's processor can typically send out within a single clock cycle of operation.

Impact: Since more instructions undergo processing throughout a cycle, a more substantial IPC is an indication of enhanced efficiency. The architecture of the processor's productivity defines IPC.

c. Clocks per Instruction(CPI)

Meaning: The mean count of clock cycles needed to complete a single instruction.

Impact: Since fewer cycles are required for each instruction, lower CPI translates into better performance. Processor design and instruction set architecture (ISA) are two examples of the variables that affect CPI.

d. Throughput

Definition: the quantity of work a processor can finish in a certain amount of time; typically expressed in transactions per second (TPS) or instructions per second (IPS).

Impact: Increased throughput is correlated with improved performance, particularly in processors with multiple cores or threads.

e. Latency (Execution Time)

Definition: the span of time needed to finish one task or instruction from beginning to end.

Impact: For real-time applications, quicker response times are essential, and lower latency translates into this.

f. Power Consumption

Definition: A processor's power consumption, expressed in watts (W) usually.

Impact: Energy efficiency requires lower power consumption, particularly in portable and battery-operated devices.

g. Thermal Design Power (TDP)

Definition: The highest possible temperature result, measured in watts, that a processor is expected generates at an usual load.

Impact: TDP has an impact on the system's overall design and cooling requirements. In order to preserve system longevity and stability, lower TDP is recommended.

h. Performance per Watt

Definition: A measurement of energy efficiency that shows how much is accomplished for every watt of power used.

Impact: Better energy efficiency is indicated by higher performance per watt, which is significant for both environmental and performance reasons.

i. Die Size

Definition: The processor die's actual size, expressed in square millimeters (mm^2).

Impact: Although smaller die sizes may result in increased yields and cheaper manufacturing costs, they may also have an effect on power and heat dissipation.

j. Transistor Count

Definition: The number of transistors in a processor.

Impact: Higher transistor counts can lead to better performance and more features but also increase complexity and manufacturing costs.

Benchmarking Tools:

Benchmarking tools are software applications and suites used to measure and evaluate the performance of processors and computing systems. These tools simulate various workloads and tasks to provide a comprehensive assessment of performance metrics. Here are some commonly used benchmarking tools mentioned in the documents:

a. SPEC CPU

   Description: The SPEC CPU benchmarks assess a processor's integer and floating-point operations performance. They are created by the Standard Performance Evaluation Corporation (SPEC).

   Use: Commonly employed to compare various processors' performance under set parameters.

b. Geekbench

   Description: A cross-platform benchmarking tool that measures the performance of a processor's single-core and multi-core capabilities.

   Usage: Popular for providing quick and comprehensive performance metrics for both desktop and mobile processors.

c. Pass Mark

   Description: A suite of benchmarks that test various aspects of CPU performance, including integer, floating-point, and encryption operations.

   Usage: Used for comparing the performance of different processors across a range of computational tasks.

d. Cinebench

   Description: Based on Maxon's Cinema 4D software, Cinebench measures the performance of a processor in rendering 3D scenes.

   Usage: Commonly used in the creative industry to evaluate the performance of processors in graphics-intensive tasks.

e. Linpack

   Description: A benchmark that measures a system's floating-point computing power by solving a dense system of linear equations.

   Usage: Often used in high-performance computing (HPC) to assess the capabilities of supercomputers and clusters.

f. PC Mark

   Description: A comprehensive benchmarking tool that measures the overall performance of a system, including CPU, GPU, memory, and storage.

   Usage: Suitable for evaluating the performance of complete systems in everyday computing tasks.

g. 3DMark

   Description: A benchmarking tool focused on gaming and graphics performance, measuring CPU and GPU capabilities in rendering 3D graphics.

   Usage: Widely used by gamers and hardware enthusiasts to evaluate the performance of gaming systems.

h. SiSoftware Sandra

   Description: A diagnostic and benchmarking tool that provides detailed information about a system's hardware and performance.

   Usage: Used for in-depth analysis and comparison of various system components, including processors.

By using these metrics and tools, designers can optimize processor architectures for better performance, efficiency, and cost-effectiveness, while consumers and professionals can make informed decisions when selecting processors for their computing needs.